Inaccessible Entropy I: Inaccessible Entropy Generators and Statistically Hiding Commitments from One-Way Functions*

Working Draft: Please Do Not Distribute

Iftach Haitner† Omer Reingold‡ Salil Vadhan§ Hoeteck Wee¶

June 22, 2020

Abstract

We put forth a new computational notion of entropy, measuring the (in)feasibility of sampling high-entropy strings that are consistent with a given generator. Specifically, the i'th output block of a generator G has *accessible entropy* at most k if the following holds: when conditioning on its prior coin tosses, no polynomial-time strategy $\widetilde{\mathsf{G}}$ can generate valid output for G's i'th output block with entropy greater than k. A generator has *inaccessible entropy* if the total accessible entropy (summed over the blocks) is noticeably smaller than the real entropy of G's output.

As an application of the above notion, we improve upon the result of Haitner, Nguyen, Ong, Reingold, and Vadhan [Sicomp '09], presenting a much simpler and more efficient construction of statistically hiding commitment schemes from arbitrary one-way functions.

Keywords: computational complexity; cryptography; commitment schemes; one-way functions; computational entropy

*A preliminary version, with a different notion of accessible entropy, appeared in [14].

†School of Computer Science, Tel Aviv University. E-mail: iftachh@cs.tau.ac.il. Supported by ERC starting grant 638121 and US-Israel BSF grant 2010196. Director of the Check Point Institute for Information Security.

‡Computer Science Department, Stanford University, reingold@stanford.edu. Supported by US-Israel BSF grant 2006060.

§School of Engineering and Applied Sciences and Center for Research on Computation and Society, Harvard University. E-mail: salil@eecs.harvard.edu. Work done in part while visiting U.C. Berkeley, supported by the Miller Institute for Basic Research in Science and a Guggenheim Fellowship. Also supported by NSF grant CNS-0831289 and US-Israel BSF grant 2006060.

¶Queens College, CUNY. E-mail: hoeteck@cs.qc.cuny.edu. Part of this work was done while a post-doc at Columbia University, supported in part by NSF Grants CNS-0716245 and SBE-0245014.

Contents

1 Introduction

Computational analogues of information-theoretic notions have given rise to some of the most interesting phenomena in the theory of computation. For example, *computational indistinguishability*, a computational analogue of statistical indistinguishability introduced by Goldwasser and Micali [9], enabled the bypassing of Shannon [29]'s impossibility results on perfectly secure encryption [29] and provided the basis for the computational theory of pseudorandomness [4, 33]. *Pseudoentropy*, a computational analogue of entropy introduced by Håstad, Impagliazzo, Levin, and Luby [19], was the key to their fundamental result that established the equivalence of pseudorandom generators and one-way functions and has become a basic concept in complexity theory and cryptography. The above notions were further refined in [17, 30], leading to to much simpler and more efficient constructions of pseudorandom generators based on one-way functions.

In this work, we introduce another computational analogue of entropy we call *accessible entropy*. We use this notion to build a simpler construction of statistically hiding and computationally binding commitment schemes from arbitrary one-way functions, a construction that is significantly simpler and more efficient than the previous construction of Haitner et al. [13]. Before describing accessible entropy (and the complementary notion of *inaccessible entropy*), we review the standard information-theoretic notion of entropy and the computational notion of pseudoentropy [19].

1.1 Entropy and Pseudoentropy

Recall that the *entropy* of a random variable X is defined to be $\mathrm{H}(X) := \mathsf{E}_{x \xleftarrow{R} X}\left[\log \frac{1}{\mathsf{Pr}[X=x]}\right]$, which measures the number of "bits of randomness" in X (on average). We will refer to $\mathrm{H}(X)$ as the *real entropy* of X to contrast with the computational analogues that we study.

Håstad et al. [19] define the *pseudoentropy* of a random variable X to be (at least) k if there exists a random variable Y of entropy (at least) k such that X and Y are computationally indistinguishable. Pseudoentropy is interesting and useful since, assuming one-way functions exist, there exist random variables whose pseudoentropy is larger than their real entropy. For example, the output of a pseudorandom generator $\mathsf{G}\colon \{0,1\}^{\ell} \mapsto \{0,1\}^{n}$ on a uniformly random seed has entropy at most ℓ, but has pseudoentropy n (by definition). Håstad et al. [19] proved that from *any* efficiently samplable distribution X whose pseudoentropy is noticeably larger than its real entropy, it is possible to construct a pseudorandom generator. By showing, in addition, how to construct such a distribution X from any one-way function, Håstad et al. [19] effectively proved their theorem that the existence of one-way functions implies the existence of pseudorandom generators.

Notions of pseudoentropy as above are only useful as a *lower* bound on the "computational entropy" in a distribution. Indeed, it can be shown that *every* distribution on $\{0,1\}^{n}$ is computationally indistinguishable from a distribution of entropy at most $\operatorname{polylog} n$. While several other computational analogues of entropy have been studied in the literature (cf., [1]), all are meant to capture the idea that a distribution "behaves like" one of higher entropy. In this paper, we explore a way in which a distribution can "behave like" one of much *lower* entropy.

1.2 Inaccessible Entropy

We begin by defining accessible entropy, from which we can derive inaccessible entropy. The notion of *accessible entropy* presented below is useful as an *upper* bound on computational entropy. We motivate the idea of accessible entropy with an example. Let G be the following two-block generator:

Algorithm 1.1 (Generator G)**.**

Let $m \ll n$ and let $\mathcal{H} = \{h\colon \{0,1\}^n \mapsto \{0,1\}^m\}$ be a family of collision-resistant hash functions.[1]

On public parameter $h \xleftarrow{R} \mathcal{H}$:

1. *Sample $x \xleftarrow{R} \{0,1\}^n$.*
2. *Output $y = h(x)$.*
3. *Output x.*

Information-theoretically, conditioned on h and its first output block y, the second output block of G (namely x) has entropy at least $n - m$. This is since $(h, y = h(x))$ reveals at most m bits of information about x. The collision-resistance property of h, however, implies that given the *state* of G after it outputs its first block y, there is at most one consistent value of x that can be computed in polynomial time with non-negligible probability. (Otherwise, we would be able to find two distinct messages $x \neq x'$ such that $h(x) = h(x')$.) This holds even if G is replaced by any polynomial-time cheating strategy $\widetilde{\mathsf{G}}$. Thus, there is "real entropy" in x (conditioned on h and the first output of G), but it is "computationally inaccessible" to $\widetilde{\mathsf{G}}$, for which the entropy of x is effectively 0.

We generalize this basic idea to consider both the real and accessible entropy accumulated over several blocks of a generator. Consider an *m-block generator* $G\colon \{0,1\}^n \mapsto (\{0,1\}^*)^m$, and let $(Y_1, \ldots, Y_m)$ be random variables denoting the m output blocks generated by applying G over randomness U_n (no public parameters are given). We define the *real entropy* of G as $\mathrm{H}(G(U_n))$, the Shannon entropy of $G(U_n)$, which is equal to

$$\sum_{i \in [m]} \mathrm{H}(Y_i \mid Y_{<i})$$

for $\mathrm{H}(X \mid Y) = \mathrm{E}_{y \xleftarrow{R} Y}[\mathrm{H}(X \mid_{Y=y})]$, which is the standard notion of (Shannon) conditional entropy, and $Y_{<i} = Y_1 \ldots, Y_{i-1}$.

To define *accessible entropy*, consider the following PPT (probabilistic polynomial-time) cheating algorithm $\widetilde{\mathsf{G}}$: before outputting the i'th block, it tosses some fresh random coins r_i, and uses them to output a string y_i. We restrict our attention to *G-consistent* (adversarial) generators — $\widetilde{\mathsf{G}}$'s output is always in the support of G (though it might be distributed differently). Now, let $(R_1, Y_1, \ldots, R_m, Y_m)$ be random variables corresponding to a random execution of $\widetilde{\mathsf{G}}$. We define the *accessible entropy* achieved by $\widetilde{\mathsf{G}}$ to be

$$\sum_{i \in [m]} \mathrm{H}(Y_i \mid Y_{<i}, R_{<i}) = \sum_{i \in [m]} \mathrm{H}(Y_i \mid R_{<i}).$$

Namely, we compute the entropy conditioned not just on the previous output blocks $Y_{<i}$ (which are determined by $R_{<i}$), as done when computing the real entropy of G, but also on the local state of $\widetilde{\mathsf{G}}$ prior to outputting the i'th block (which without loss of generality equals its coin tosses $R_{<i}$). For a given G, we define its *accessible entropy* as the maximal accessible entropy achieved by a

[1] Given $h \xleftarrow{\mathrm{R}} \mathcal{H}$, it is infeasible to find distinct $x, x' \in \{0,1\}^n$ with $h(x) = h(x')$.

G-consistent, polynomial-time generator $\widetilde{\mathsf{G}}$.[2] Thus, in contrast to pseudoentropy, accessible entropy is useful for expressing the idea that the "computational entropy" in a distribution is *smaller* than its real entropy. We refer to the difference (real entropy) − (accessible entropy) as the *inaccessible entropy* of the protocol.

It is important to note that if we put no restrictions on the computational power of a G-consistent $\widetilde{\mathsf{G}}$, then its accessible entropy can always be as high as the real entropy of G; to generate its i'th block y_i, $\widetilde{\mathsf{G}}$ samples x uniformly at random from the set $\{x' \colon \mathsf{G}(x')_1 = y_1, \ldots, \mathsf{G}(x')_{i-1} = y_{i-1}\}$. This strategy, however, is not always possible for a computationally bounded $\widetilde{\mathsf{G}}$.

The collision resistance example given earlier provides evidence that when allowing public parameters, there are efficient generators whose computationally accessible entropy is much smaller than their real Shannon entropy. Indeed, the real entropy of the generator we considered above is n (namely, the total entropy in x), but its accessible entropy is at most $m + \mathrm{neg}(n) \ll n$, where m is the output length of the collision-resistant hash function.

1.3 Constructing an Inaccessible Entropy Generator from One-Way Functions

It turns out that we do not need to assume collision resistance or use pubic parameters to have a generator whose real entropy is significantly larger than its accessible entropy; any one-way function can be used to construct an inaccessible entropy generator (without public parameters). In particular, we prove the following result:

Theorem 1.2 (Inaccessible entropy generator from one-way functions, informal)**.** *For a function $f \colon \{0,1\}^n \mapsto \{0,1\}^n$, let G be the $(n+1)$-block generator defined by*

$$\mathsf{G}(x) = (f(x)_1, f(x)_2, \ldots, f(x)_n, x).$$

Assuming f is a one-way function, then the accessible entropy of G is at most $n - \log n$. Since the real entropy of $G(U_n)$ is n, it follows that G has $\log n$ bits of inaccessible entropy.

Interestingly, the definition of G used in the above theorem is the same as the construction of a next-block pseudoentropy generator from a one-way function used by Vadhan and Zheng [30], except that we have broken $f(x)$, other than x, into one-bit blocks.

To show that G has accessible entropy at most $n - \log n$, we show that the existence of an efficient algorithm whose accessible entropy is too high yields an efficient inverter for f (contradicting its one-wayness). Assume for simplicity that f is a permutation and that there exists an efficient algorithm whose accessible entropy is as high as the real entropy of G (i.e., n). Let Y_i denote the i'th output block of $\widetilde{\mathsf{G}}$ and let R_i be the coins used by $\widetilde{\mathsf{G}}$ to produce Y_i. By assumption, $\sum_{i \in [n]} \mathrm{H}(Y_i \mid R_{<i}) = \mathrm{H}(f(U_n)) = n$. Since $\mathrm{H}(Y_i \mid R_{<i}) \leq 1$ for every $i \in [n]$, it follows that $\mathrm{H}(Y_i \mid R_{<i}) = 1$ for every $i \in [n]$.

The above observation yields the following strategy to invert f on input y: for $i = 1$ to n, keep sampling values for R_i until the induced value of Y_i is equal to y_i. The high accessible entropy of each output block of $\widetilde{\mathsf{G}}$ yields that the expected number of samples per output bit is two. Therefore, with high probability, after a polynomial number of samples the above process terminates successfully,

[2] The above informal definitions are simplified or restricted compared to our actual definitions in several ways. In particular, for some of our reductions it is beneficial to work with real *min*-entropy and/or accessible *max*-entropy rather than real and accessible Shannon entropy as defined above, and formulating conditional versions of these measures is a bit more delicate.

making the first n output bits of $\widetilde{\mathsf{G}}$ to be equal to y. This implies an efficient inverter for f, since when the above process is done, the definition of $\widetilde{\mathsf{G}}$ yields that its "justification" for the n'th output block is the preimage of y.

1.4 Statistically Hiding Commitment from One-way Functions

As an application of the above accessible entropy notion, we present a much simpler and more efficient construction of statistically hiding commitment schemes from arbitrary one-way functions. Our construction builds such a commitment from a generator with noticeable accessible entropy whose existence is guaranteed by Theorem 1.2. The resulting scheme conceptually unifies the construction of statistically *hiding* commitments from one-way functions with the construction of statistically *binding* commitments from one-way functions (the latter being due to [19, 26]): the first step of both constructions is to obtain a gap between real entropy and "computational entropy" (pseudoentropy in the case of statistical binding and accessible entropy in the case of statistical hiding), which is then amplified by repetitions and finally combined with various forms of hashing.

We start by describing the above mentioned commitment schemes, and then explain how to construct them from an inaccessible entropy generator.

1.4.1 Commitment Schemes

A *commitment scheme* is the cryptographic analogue of a safe. It is a two-party protocol between a *sender* S and a *receiver* R that consists of two stages. The *commit stage* corresponds to putting an object in a safe and locking it. In this stage, the sender "commits" to a private message m. The *reveal stage* corresponds to unlocking and opening the safe. In this stage, the sender "reveals" the message m and "proves" that it was the value committed to in the commit stage (without loss of generality, by revealing coin tosses consistent with m and the transcript of the commit stage).

Commitment schemes have two security properties. The *hiding* property informally states that at the end of the commit stage, an adversarial receiver has learned nothing about the message m, except with negligible probability. The *binding* property states that after the commit stage, an adversarial sender cannot output valid openings for two distinct messages, except with negligible probability. Both of these security properties come in two flavors — *statistical*, where we require security even against a computationally unbounded adversary, and *computational*, where we only require security against feasible (e.g. polynomial-time) adversaries.

Statistical security is preferable to computational security, but it is easy to see that commitment schemes that are both statistically hiding and statistically binding do not exist. Instead we have to settle for one of the two properties being statistical and the other being computational. Statistically binding (and computationally hiding) commitments have been well understood for a long time. Indeed, Naor [26] showed how to build a two-message statistically binding commitment using any pseudorandom generator; thus, in combination with the construction of pseudorandom generators from any one-way function [19], we obtain two-message statistically binding commitments from the minimal assumption that one-way functions exist.

In contrast, our understanding of *statistically hiding* commitments has lagged behind. Haitner et al. [13] have shown that statistically hiding commitment schemes can be constructed from any one-way function. Their construction, however, is very complicated and inefficient. In this paper, we show that these two types of commitments are closely connected to the notion of inaccessible entropy, that is, with protocols having a gap between real entropy and accessible entropy.

Consider a statistically hiding commitment scheme in which the sender commits to a message of length k, and suppose we run the protocol with the message m chosen uniformly at random in $\{0,1\}^k$. Then, by the statistical hiding property, the *real entropy* of the message m after the commit stage is $k - \mathrm{neg}(n)$. On the other hand, the computational binding property states that the *accessible entropy* of m after the commit stage is at most $\mathrm{neg}(n)$. Our main technical contribution is the converse to the above observation.

1.4.2 Statistically Hiding Commitment Schemes from Inaccessible Entropy Generators

Theorem 1.3 (inaccessible entropy to commitment, informal)**.** *Assume there exists an m-block efficient generator with real entropy n and accessible entropy at most $k(1-\delta)$. Then there exists a $\Theta(m^2/\delta^2)$-round statistically hiding commitment scheme. Moreover, if we allow the protocol to use nonuniform advice, the round complexity is reduced to $\Theta(m/\delta)$.*

We prove Theorem 1.3 via few modular steps:

Entropy equalization Using sequential repetition with a "random offset", we convert the generator into one for which we *know* the real entropy in each block (rather than just knowing the total entropy) and there remains a noticeable gap between the real entropy and the accessible entropy.

Gap amplification Taking the direct product of the generator (i.e., invoking it many times in parallel) has the effect of (a) converting the real entropy to real *min*-entropy, and (b) amplifying the gap between the real entropy and accessible entropy.

Constructing the commitment scheme By applying a constant-round hashing protocol in each round (based on the interactive hashing protocol of [7] and universal one-way hash functions [27, 28]), we obtain a receiver public-coin *weakly binding* statistically hiding commitment scheme. The latter commitment is then amplified into a full-fledged commitment using parallel repetition.

Statistically hiding commitments from one-way functions. Combining Theorems 1.2 and 1.3 yields that the minimal assumption that one-way functions exist implies the existence of statistically hiding commitment schemes, reproving [13].

Theorem 1.4 (Statistically hiding commitments from one-way functions, informal)**.** *Assume one-way functions exist. Then there exists an $\Theta(n^2/\log n^2)$-round statistically hiding commitment scheme, where n is the input length of the one-way function. Moreover, if we allow the protocol to use nonuniform advice, the round complexity is reduced to obtain $\Theta(n/\log n)$.*

The above improves upon the protocol of [13], which has a large unspecified polynomial number of rounds, and the nonuniform variant meets the lower bound of [18] for "black-box constructions" (such as the one we used to prove the theorem).

1.5 Related Work

A preliminary version of Haitner et al. [14] uses a more general, and more complicated, notion of accessible entropy in which the accessible entropy of *protocols* rather than generators is measured.

This latter notion is used in that paper to show that if NP has constant-round interactive proofs that are black-box zero knowledge under parallel composition, then there exist constant-round statistically hiding commitment schemes. A subsequent work of Haitner et al. [15] uses a simplified version of accessible entropy to present a simpler and more efficient construction of *universal one-way functions* from any one-way function. One of the two inaccessible entropy generators considered in [15], they considered for this construction, is very similar to the construction of inaccessible entropy generators discussed above. The notion of inaccessible entropy, of the simpler variant appearing in that work, is in a sense implicit in the work of Rompel [28], who first showed how to base universal one-way functions on any one-way function. Very recently, Bitansky et al. [3] used the reduction given in this paper from an inaccessible entropy generator to statistically hiding commitment, to construct constant-round statistically hiding commitments from *distributional collision resistance hash functions*, a relaxation of collision resistance hash functions known to exist assuming average hardness of the class SZK. Finally, a simplified presentation of the definitions and reductions appearing in this paper can be found in [11].

Paper Organization

Standard notations and definitions are given in Section 2; we also give there several useful facts about the conditional entropy of sequences of random variables. Formal definitions of the real and accessible entropy of a generator are given in Section 3. In Section 4, we show how to construct inaccessible entropy generators from one-way functions. In Section 5, we develop tools to manipulate the real and accessible entropy of such generators. In Section 6, we construct a statistically hiding commitment scheme from a generator that has a (noticeable) gap between its real and accessible entropy, and then use this reduction together with the results of Sections 4 and 5 to construct a statistically hiding commitment scheme from one-way functions. Finally, in Section 7 we prove that the existence of inaccessible entropy generators implies that of one-way functions (namely, we prove the converse of the main result of Section 4).

2 Preliminaries

2.1 Notation

We use calligraphic letters to denote sets, upper-case for random variables, lower-case for values, bold-face for vectors, and sans serif for algorithms (i.e., Turing machines). For $n \in \mathbb{N}$, let $[n] = \{1, \ldots, n\}$. For vector $\boldsymbol{y} = (y_1, \ldots, y_n)$ and $\mathcal{J} \subseteq [n]$, let $\boldsymbol{y}_{\mathcal{J}} = (y_{i_1}, \ldots, y_{i_{|\mathcal{J}|}})$, where $i_1 < \ldots < i_{|\mathcal{J}|}$ are the elements of $\mathcal{J}$. Let $\boldsymbol{y}_{<j} = \boldsymbol{y}_{[j-1]} = (y_1, \ldots, y_{j-1})$ and $\boldsymbol{y}_{\leq j} = \boldsymbol{y}_{[j]} = (y_1, \ldots, y_j)$. Both notations naturally extend to an ordered list of elements that is embedded in a larger vector (i.e., given $(a_1, b_1, \ldots, a_n, b_n)$, $a_{<3}$ refers to the vector (a_1, a_2)). Let poly denote the set of all positive polynomials. A function $\nu \colon \mathbb{N} \mapsto [0,1]$ is *negligible*, denoted $\nu(n) = \mathrm{neg}(n)$, if $\nu(n) < 1/p(n)$ for every $p \in \mathrm{poly}$ and large enough n.

2.2 Random Variables

Let X and Y be random variables taking values in a discrete universe $\mathcal{U}$. We adopt the convention that, when the same random variable appears multiple times in an expression, all occurrences refer to the same instantiation. For example, $\mathsf{Pr}[X = X]$ is 1. For an event E, we write $X|_E$ to denote the

random variable X conditioned on E. We let $\mathsf{Pr}_{X|Y}[x|y]$ stand for $\mathsf{Pr}[X = x \mid Y = y]$. The *support* of a random variable X, denoted $\mathrm{Supp}(X)$, is defined as $\{x\colon \mathsf{Pr}[X = x] > 0\}$. Let U_n denote a random variable that is uniform over $\{0,1\}^n$. For $t \in \mathbb{N}$, let $X^{(t)} = (X^1, \ldots, X^t)$, where $X^1, \ldots, X^t$ are independent copies of X.

We write $X \equiv Y$ to indicate that X and Y are identically distributed. We write $\Delta(X,Y)$ to denote the *statistical difference* (also known as variation distance) between X and Y, i.e.,

$$\Delta(X,Y) = \max_{T \subseteq \mathcal{U}} \left|\mathsf{Pr}[X \in T] - \mathsf{Pr}[Y \in T]\right|.$$

If $\Delta(X,Y) \leq \varepsilon$ [resp., $\Delta(X,Y) > \varepsilon$], we say that X and Y are *ε-close* [resp., ε-far]. Two random variables $X = X(n)$ and $Y = Y(n)$ are *statistically indistinguishable*, denoted $X \approx_s Y$, if for any unbounded algorithm D, it holds that $|\mathsf{Pr}[\mathsf{D}(1^n, X(n)) = 1] - \mathsf{Pr}[\mathsf{D}(1^n, Y(n)) = 1]| = \mathrm{neg}(n)$.[3] Similarly, X and Y are computationally indistinguishable, denoted $X \approx Y]$, if $|\mathsf{Pr}[\mathsf{D}(1^n, X(n)) = 1] - \mathsf{Pr}[\mathsf{D}(1^n, Y(n)) = 1]| = \mathrm{neg}(n)$ for every PPT D.

The KL-divergence (also known as Kullback-Leibler divergence and relative entropy) between two distributions P, Q over a discrete domain $\mathcal{X}$ is defined by

$$\mathsf{D}\big(P \,\|\, Q\big) := \sum_{x \in \mathcal{X}} P(x) \log \frac{P(x)}{Q(x)} = \mathsf{E}_{x \xleftarrow{\mathrm{R}} P} \log \frac{P(x)}{Q(x)},$$

letting $0 \cdot \log \frac{0}{0} = 0$, and if there exists $x \in \mathcal{X}$ such that $P(x) > 0 = Q(x)$ then $\mathsf{D}\big(P \,\|\, Q\big) := \infty$.

2.3 Entropy Measures

We refer to several measures of entropy. The relation and motivation of these measures is best understood by considering a notion that we will refer to as the sample-entropy: for a random variable X and $x \in \mathrm{Supp}(X)$, the *sample-entropy* of x with respect to X is the quantity

$$\mathrm{H}_X(x) := \log \tfrac{1}{\mathsf{Pr}[X=x]},$$

letting $\mathrm{H}_X(x) = \infty$ for $x \notin \mathrm{Supp}(X)$, and $2^{-\infty} = 0$.

The sample-entropy measures the amount of "randomness" or "surprise" in the specific sample x, assuming that x has been generated according to X. Using this notion, we can define the *Shannon entropy* $\mathrm{H}(X)$ and *min-entropy* $\mathrm{H}_\infty(X)$ as follows:

$$\mathrm{H}(X) := \mathop{\mathrm{E}}_{x \xleftarrow{\mathrm{R}} X}[\mathrm{H}_X(x)],$$
$$\mathrm{H}_\infty(X) := \min_{x \in \mathrm{Supp}(X)} \mathrm{H}_X(x).$$

The *collision probability* of X is defined by

$$\mathrm{CP}(X) := \sum_{x \in \mathrm{Supp}(X)} \mathsf{Pr}_X[x]^2 = \mathsf{Pr}_{(x,x') \xleftarrow{\mathrm{R}} X^2}\left[x = x'\right]$$

and its *Rényi entropy* is defined by

$$\mathrm{H}_2(X) := -\log \mathrm{CP}(X).$$

[3] This is equivalent to to requiring that $\Delta(X(n), Y(n)) = \mathrm{neg}(n)$.

We will also discuss the *max-entropy* $\mathrm{H}_0(X) := \log|\mathrm{Supp}(X)|$. The term "max-entropy" and its relation to the sample-entropy will be made apparent below.

It can be shown that $\mathrm{H}_\infty(X) \le \mathrm{H}_2(X) \le \mathrm{H}(X) \le \mathrm{H}_0(X)$ with each inequality being an equality if and only if X is flat (uniform on its support). Thus, saying that $\mathrm{H}_\infty(X) \ge k$ is a strong way of saying that X has "high entropy" and $\mathrm{H}_0(X) \le k$ a strong way of saying that X has "low entropy".

The following fact quantifies the probability that the sample-entropy is larger than the max-entropy.

Lemma 2.1. *For random variable X it holds that*

1. $\mathsf{E}_{x \xleftarrow{R} X}\left[2^{\mathrm{H}_X(x)}\right] = |\mathrm{Supp}(X)|$.
2. $\mathsf{Pr}_{x \xleftarrow{R} X}\left[\mathrm{H}_X(x) > \log\frac{1}{\varepsilon} + \mathrm{H}_0(X)\right] < \varepsilon$, *for any* $\varepsilon > 0$.

Proof. For the first item, compute

$$\begin{aligned}\mathsf{E}_{x \xleftarrow{R} X}\left[2^{\mathrm{H}_X(x)}\right] &= \sum_{x \in \mathrm{Supp}(X)} 2^{-\mathrm{H}_X(x)} \cdot 2^{\mathrm{H}_X(x)} \\ &= \sum_{x \in \mathrm{Supp}(X)} 1 \\ &= |\mathrm{Supp}(X)|\,.\end{aligned}$$

The second item follows by the first item and Markov inequality.

$$\begin{aligned}\mathsf{Pr}_{x \xleftarrow{R} X}\left[\mathrm{H}_X(x) > \log\frac{1}{\varepsilon} + \mathrm{H}_0(X)\right] &= \mathsf{Pr}_{x \xleftarrow{R} X}\left[2^{\mathrm{H}_X(x)} > \frac{1}{\varepsilon} \cdot |\mathrm{Supp}(X)|\right] \\ &< \varepsilon.\end{aligned}$$

□

The following fact quantifies the contribution of unlikely events with high sample-entropy to the overall entropy of a random variable.

Lemma 2.2. *Let X be a random variable with $\mathsf{Pr}_{x \xleftarrow{R} X}[\mathrm{H}_X(x) > k] \le \varepsilon \in [0,1]$. Then $\mathrm{H}(X) \le (1-\varepsilon)k + \varepsilon \cdot (\mathrm{H}_0(X) - \log\varepsilon) \le k + \varepsilon \cdot \mathrm{H}_0(X) + 1$.*

Proof. Let $\mathcal{Y} = \{x \in \mathrm{Supp}(X) \colon \mathrm{H}_X(x) > k\}$ and let $Y = X|_{X \in \mathcal{Y}}$. Note that

$$\mathrm{H}(Y) = \mathsf{E}_{y \leftarrow Y}[\mathrm{H}_Y(y)] = \mathsf{E}_{y \leftarrow Y}[\mathrm{H}_X(y)] + \log \mathsf{Pr}_{x \xleftarrow{R} X}[\mathrm{H}_X(x) > k]\,. \tag{1}$$

Let $\varepsilon' = \mathsf{Pr}_{x \xleftarrow{R} X}[\mathrm{H}_X(x) > k]$ (hence, $\varepsilon' \le \varepsilon$). We conclude that

$$\begin{aligned}\mathrm{H}(X) &= \sum_{x \in \mathrm{Supp}(X) \setminus \mathcal{Y}} \mathsf{Pr}[X = x] \cdot \mathrm{H}_X(x) + \sum_{x \in \mathcal{Y}} \mathsf{Pr}[X = x] \cdot \mathrm{H}_X(x) \\ &\le (1-\varepsilon')k + \varepsilon' \cdot \sum_{x \in \mathcal{Y}} \mathsf{Pr}[Y = x] \cdot \mathrm{H}_X(x) \\ &= (1-\varepsilon')k + \varepsilon' \cdot (\mathrm{H}(Y) - \log\varepsilon') \\ &\le (1-\varepsilon')k + \varepsilon' \cdot (\mathrm{H}_0(X) - \log\varepsilon') \\ &\le (1-\varepsilon)k + \varepsilon \cdot (\mathrm{H}_0(X) - \log\varepsilon) \\ &\le k + \varepsilon \cdot \mathrm{H}_0(X) + 1.\end{aligned}$$

□

Conditional entropies. We will also be interested in conditional versions of entropy. For jointly distributed random variables (X, Y) and $(x, y) \in \mathrm{Supp}(X, Y)$, we define the *conditional sample-entropy* to be $\mathrm{H}_{X|Y}(x|y) = \log \frac{1}{\mathsf{Pr}_{X|Y}[x|y]} = \log \frac{1}{\mathsf{Pr}[X=x|Y=y]}$. Then the standard *conditional Shannon entropy* can be written as

$$\mathrm{H}(X \mid Y) = \mathop{\mathrm{E}}_{(x,y)\overset{\mathrm{R}}{\leftarrow}(X,Y)} \left[\mathrm{H}_{X|Y}(x \mid y)\right] = \mathop{\mathrm{E}}_{y\overset{\mathrm{R}}{\leftarrow}Y} \left[\mathrm{H}(X|_{Y=y})\right] = \mathrm{H}(X, Y) - \mathrm{H}(Y).$$

The following known lemma states that conditioning on a "short" variable is unlikely to change the sample-entropy significantly.[4]

Lemma 2.3. *Let X and Y be random variables, let $k = \mathrm{H}_\infty(X)$, and let $\ell = \mathrm{H}_0(Y)$. Then, for any $t > 0$, it holds that*

$$\mathsf{Pr}_{(x,y)\overset{R}{\leftarrow}(X,Y)} \left[\mathrm{H}_{X|Y}(x|y) < k - \ell - t\right] < 2^{-t}.$$

Proof. For $y \in \mathrm{Supp}(Y)$, let $\mathcal{X}_y = \left\{x \in \mathrm{Supp}(X) \colon \mathrm{H}_{X|Y}(x|y) < k - \ell - t\right\}$. We have $|\mathcal{X}_y| < 2^{k-\ell-t}$. Hence, $\left|\mathcal{X} = \bigcup_{y\in\mathrm{Supp}(Y)} \mathcal{X}_y\right| < 2^\ell \cdot 2^{k-\ell-t} = 2^{k-t}$. It follows that

$$\mathsf{Pr}_{(x,y)\overset{\mathrm{R}}{\leftarrow}(X,Y)} \left[\mathrm{H}_{X|Y}(x|y) < k - \ell - t\right] \leq \mathsf{Pr}_{(x,y)\overset{\mathrm{R}}{\leftarrow}(X,Y)} \left[x \in \mathcal{X}\right] < 2^{-k} \cdot 2^{k-t} = 2^{-t}.$$

□

Smoothed entropies. The following lemma will allow us to think of a random variable X whose sample-entropy is high, with high probability, as if it has high min-entropy (i.e., as if its sample-entropy function is "smoother", with no peaks).

Lemma 2.4. *Let X and Y be random variables and let $\varepsilon > 0$.*

1. *Suppose $\mathsf{Pr}_{x\overset{R}{\leftarrow}X} \left[\mathrm{H}_X(x) \geq k\right] \geq 1 - \varepsilon$. Then X is ε-close to a random variable X' with $\mathrm{H}_\infty(X') \geq k$.*
2. *Suppose $\mathsf{Pr}_{(x,y)\overset{R}{\leftarrow}(X,Y)} \left[\mathrm{H}_{X|Y}(x|y) \geq k\right] \geq 1 - \varepsilon$. Then (X, Y) is ε-close to a random variable (X', Y') with $\mathrm{H}_{X'|Y'}(x|y) \geq k$ for any $(x, y) \in \mathrm{Supp}(X', Y')$. Further, Y' and Y are identically distributed.*

Proof. For the first item, we modify X on an ε fraction of the probability space (corresponding to when X takes on a value x such that $\mathrm{H}_X(x) \geq k$) to bring all probabilities to be smaller than or equal to 2^{-k}.

The second item is proved via similar means, while when changing (X, Y), we do so without changing the "Y" coordinate. □

[4] We could save a few bits, as compared to the result below, by considering the *average* sample-entropy induced by the conditioning, and in particular the *average min-entropy* of the variable [8]. Doing so, however, will only reduce the running time and communication size of our commitment scheme by a constant, so we preferred to stay with the simpler statement below.

Flattening Shannon entropy. It is well known that the Shannon entropy of a random variable can be converted to min-entropy (up to small statistical distance) by taking independent copies of this variable.

Lemma 2.5 ([32], Theorem 3.14)**.** *Let X be a random variable taking values in a universe $\mathcal{U}$, let $t \in \mathbb{N}$, and let $0 < \varepsilon \leq 1/e^2$. Then with probability at least $1 - \varepsilon$ over $x \xleftarrow{R} X^{(t)}$,*

$$\mathrm{H}_{X^{(t)}}(x) - t \cdot \mathrm{H}(X) \geq -O\left(\sqrt{t \cdot \log \tfrac{1}{\varepsilon}} \cdot \log(|\mathcal{U}| \cdot t)\right).$$

We will make use of the following "conditional variant" of Lemma 2.5:

Lemma 2.6. *Let X and Y be jointly distributed random variables where X takes values in a universe $\mathcal{U}$, let $t \in \mathbb{N}$, and let $0 < \varepsilon \leq 1/e^2$. Then with probability at least $1 - \varepsilon$ over $(x, y) \leftarrow (X', Y') = (X, Y)^{(t)}$,*

$$\mathrm{H}_{X'|Y'}(x \mid y) - t \cdot \mathrm{H}(X \mid Y) \geq -O\left(\sqrt{t \cdot \log \tfrac{1}{\varepsilon}} \cdot \log(|\mathcal{U}| \cdot t)\right).$$

Proof. Follows the same line as the proof of Lemma 2.5, by considering the random variable $\mathrm{H}_{X|Y}(X|Y)$ instead of $\mathrm{H}_X(X)$. □

Sequence of random variables. For measuring the real and accessible entropy of a generator, we will measure the entropy of a subset of random variables, conditioned on the previous elements in the sequence, The following lemma generalizes Lemma 2.2 to settings that come up naturally when measuring the accessible entropy of a generator (as we do in Section 5).

Definition 2.7. *For a t-tuple random variable $\boldsymbol{X} = (X_1, \ldots, X_t)$, $\boldsymbol{x} \in \mathrm{Supp}(X)$ and $\mathcal{J} \subseteq [t]$, let*

$$\mathrm{H}_{\boldsymbol{X},\mathcal{J}}(\boldsymbol{x}) = \sum_{i \in \mathcal{J}} \mathrm{H}_{\boldsymbol{X}_i|\boldsymbol{X}_{<i}}(\boldsymbol{x}_i|\boldsymbol{x}_{<i}).$$

The following fact is immediate by the chain rule.

Proposition 2.8. *Let $\boldsymbol{X} = (X_1, \ldots, X_t)$ be a sequence of random variables and let $\mathcal{J} \subseteq [t]$. Then, $\mathsf{E}_{\boldsymbol{x} \xleftarrow{R} \boldsymbol{X}}\left[\mathrm{H}_{\boldsymbol{X},\mathcal{J}}(\boldsymbol{x})\right] = \sum_{j \in \mathcal{J}} \mathrm{H}(X_j|\boldsymbol{X}_{<j}) \leq \mathrm{H}_0(\boldsymbol{X}_{\mathcal{J}})$.*

Proof. Compute

$$\begin{aligned}
\mathsf{E}_{\boldsymbol{x} \xleftarrow{R} \boldsymbol{X}}\left[\mathrm{H}_{\boldsymbol{X},\mathcal{J}}(\boldsymbol{x})\right] &= \mathsf{E}_{\boldsymbol{x} \xleftarrow{R} \boldsymbol{X}}\left[\sum_{j \in \mathcal{J}} \mathrm{H}_{X_j|\boldsymbol{X}_{<j}}(\boldsymbol{x}_j|\boldsymbol{x}_{<j})\right] \\
&= \sum_{j \in \mathcal{J}} \mathsf{E}_{\boldsymbol{x} \xleftarrow{R} \boldsymbol{X}}\left[\mathrm{H}_{X_j|\boldsymbol{X}_{<j}}(\boldsymbol{x}_j|\boldsymbol{x}_{<j})\right] \\
&= \sum_{j \in \mathcal{J}} \mathrm{H}(X_j|\boldsymbol{X}_{<j}) \\
&\leq \sum_{j \in \mathcal{J}} \mathrm{H}_0(X_j) \leq \mathrm{H}_0(\boldsymbol{X}_{\mathcal{J}}).
\end{aligned}$$

□

The following lemma generalizes Lemma 2.2 to such a sequence of random variables.

Lemma 2.9. *Let $\boldsymbol{X} = (X_1, \ldots, X_t)$ be a sequence of random variables and let $\mathcal{J} \subseteq [t]$. If $\mathsf{Pr}_{\boldsymbol{x} \xleftarrow{R} \boldsymbol{X}}[\mathrm{H}_{\boldsymbol{X},\mathcal{J}}(\boldsymbol{x}) > k] \leq \varepsilon \in [0,1]$, then $\mathsf{E}_{\boldsymbol{x} \xleftarrow{R} \boldsymbol{X}}[\mathrm{H}_{\boldsymbol{X},\mathcal{J}}(\boldsymbol{x})] \leq (1-\varepsilon)k + \varepsilon \cdot (\mathrm{H}_0(\boldsymbol{X}_{\mathcal{J}}) - \log 1/\varepsilon) \leq k + \varepsilon \cdot \mathrm{H}_0(\boldsymbol{X}_{\mathcal{J}}) + 1$.*

Proof. The proof is similar to that of Lemma 2.2. Let $\mathcal{Y} = \{\boldsymbol{x} \in \mathrm{Supp}(\boldsymbol{X})\colon \mathrm{H}_{\boldsymbol{X},\mathcal{J}}(\boldsymbol{x}) > k\}$, let $\mathbf{Y} = \boldsymbol{X}|_{\boldsymbol{X} \in \mathcal{Y}}$, and for $\boldsymbol{y}_{\leq k} \in \mathrm{Supp}(\mathbf{Y}_{<j})$ let $\varepsilon_{\boldsymbol{y}_{\leq j}} = \frac{\mathsf{Pr}[\mathbf{Y}_j = \boldsymbol{y}_j | \mathbf{Y}_{<j} = \boldsymbol{y}_{<j}]}{\mathsf{Pr}[\boldsymbol{X}_j = \boldsymbol{y}_j | \boldsymbol{X}_{<j} = \boldsymbol{y}_{<j}]}$. Compute

$$\begin{aligned}
\mathsf{E}_{\boldsymbol{y} \xleftarrow{R} \mathbf{Y}}[\mathrm{H}_{\mathbf{Y},\mathcal{J}}(\boldsymbol{y})] &= \mathsf{E}_{\boldsymbol{y} \xleftarrow{R} \mathbf{Y}}\left[\sum_{j \in \mathcal{J}} \mathrm{H}_{\mathbf{Y}_i|\mathbf{Y}_{<j}}(\boldsymbol{y}_j|\boldsymbol{y}_{<j})\right] \\
&= \mathsf{E}_{\boldsymbol{y} \xleftarrow{R} \mathbf{Y}}\left[\sum_{j \in \mathcal{J}} \mathrm{H}_{\boldsymbol{X}_j|\boldsymbol{X}_{<j}}(\boldsymbol{y}_j|\boldsymbol{y}_{<j}) + \log \frac{\mathsf{Pr}[\mathbf{Y}_j = \boldsymbol{y}_j | \mathbf{Y}_{<j} = \boldsymbol{y}_{<j}]}{\mathsf{Pr}[\boldsymbol{X}_j = \boldsymbol{y}_j | \boldsymbol{X}_{<j} = \boldsymbol{y}_{\leq j}]}\right] \\
&= \mathsf{E}_{\boldsymbol{y} \xleftarrow{R} \mathbf{Y}}\left[\sum_{j \in \mathcal{J}} \mathrm{H}_{\boldsymbol{X}_j|\boldsymbol{X}_{<j}}(\boldsymbol{y}_j|\boldsymbol{y}_{<j}) + \log \varepsilon_{\boldsymbol{y}_{\leq j}}\right] \\
&= \mathsf{E}_{\boldsymbol{y} \xleftarrow{R} \mathbf{Y}}\left[\sum_{j \in \mathcal{J}} \mathrm{H}_{\boldsymbol{X}_j|\boldsymbol{X}_{<j}}(\boldsymbol{y}_j|\boldsymbol{y}_{<j})\right] + \mathsf{E}_{\boldsymbol{y} \xleftarrow{R} \mathbf{Y}}\left[\sum_{j \in \mathcal{J}} \log \varepsilon_{\boldsymbol{y}_{\leq j}}\right], \qquad (2)
\end{aligned}$$

and note that

$$\begin{aligned}
\mathsf{E}_{\boldsymbol{y} \xleftarrow{R} \mathbf{Y}}\left[\sum_{j \in \mathcal{J}} \log \varepsilon_{\boldsymbol{y}_{\leq j}}\right] &\geq \mathsf{E}_{\boldsymbol{y} \xleftarrow{R} \mathbf{Y}}\left[\sum_{j \in [t]} \log \varepsilon_{\boldsymbol{y}_{\leq j}}\right] \qquad (3) \\
&= \mathsf{E}_{\boldsymbol{y} \xleftarrow{R} \mathbf{Y}}\left[\log \prod_j \varepsilon_{\boldsymbol{y}_{\leq j}}\right] \\
&= \mathsf{E}_{\boldsymbol{y} \xleftarrow{R} \mathbf{Y}}\left[\log \frac{\mathsf{Pr}[\mathbf{Y} = \boldsymbol{y}]}{\mathsf{Pr}[\boldsymbol{X} = \boldsymbol{y}]}\right] \\
&= \log 1/\mathsf{Pr}[\boldsymbol{X} \in \mathcal{Y}].
\end{aligned}$$

Let $\varepsilon' = \mathsf{Pr}[\boldsymbol{X} \in \mathcal{Y}]$. We conclude that

$$\begin{aligned}
\mathsf{E}_{\boldsymbol{x} \xleftarrow{R} \boldsymbol{X}}[\mathrm{H}_{\mathbf{Y},\mathcal{J}}(\boldsymbol{x})] &= \sum_{\boldsymbol{x} \in \mathrm{Supp}(\boldsymbol{X}) \setminus \mathcal{Y}} \mathsf{Pr}[\boldsymbol{X} = \boldsymbol{x}] \cdot \mathrm{H}_{\boldsymbol{X},\mathcal{J}}(\boldsymbol{x}) + \sum_{\boldsymbol{x} \in \mathcal{Y}} \mathsf{Pr}[\boldsymbol{X} = \boldsymbol{x}] \cdot \mathrm{H}_{\boldsymbol{X},\mathcal{J}}(\boldsymbol{x}) \\
&\leq (1-\varepsilon')k + \varepsilon' \cdot \sum_{\boldsymbol{x} \in \mathcal{Y}} \mathsf{Pr}[\mathbf{Y} = \boldsymbol{x}] \cdot \mathrm{H}_{\boldsymbol{X},\mathcal{J}}(\boldsymbol{x}) \\
&\leq (1-\varepsilon')k + \varepsilon' \cdot (\mathsf{E}_{\boldsymbol{y} \xleftarrow{R} \mathbf{Y}}[\mathrm{H}_{\mathbf{Y},\mathcal{J}}(\boldsymbol{y})] - \log 1/\varepsilon') \\
&\leq (1-\varepsilon')k + \varepsilon' \cdot (\mathrm{H}_0(\boldsymbol{X}_{\mathcal{J}}) - \log 1/\varepsilon') \\
&\leq (1-\varepsilon)k + \varepsilon \cdot (\mathrm{H}_0(\boldsymbol{X}_{\mathcal{J}}) - \log 1/\varepsilon) \\
&\leq k + \varepsilon \cdot \mathrm{H}_0(\boldsymbol{X}_{\mathcal{J}}) + 1.
\end{aligned}$$

□

The next two lemmas are only needed when measuring the *max* accessible entropy of a generator, as we do in Appendix A, and are not used in the main body of the paper. The first lemma generalizes Lemma 2.1 to sequence of random variables.

Lemma 2.10. *Let $\boldsymbol{X} = (X_1, \ldots, X_t)$ be a sequence of random variables and let $\mathcal{J} \subseteq [t]$. Then,*

1. $\mathsf{E}_{\boldsymbol{x} \xleftarrow{R} \boldsymbol{X}} \left[2^{\mathrm{H}_{\boldsymbol{X},\mathcal{J}}(\boldsymbol{x})}\right] \leq \mathrm{H}_0(\boldsymbol{X}_{\mathcal{J}})$.
2. $\mathsf{Pr}_{x \xleftarrow{R} X} \left[\mathrm{H}_{\boldsymbol{X},\mathcal{J}}(\boldsymbol{x}) > \log \frac{1}{\varepsilon} + \mathrm{H}_0(X_{\mathcal{J}})\right] < \varepsilon$, *for any* $\varepsilon > 0$.

Proof. The second item follows from the first one as in the proof of Lemma 2.1. We prove the first item by induction on t and $|J|$. The case $t = 1$ is immediate, so we assume for all $(t', \mathcal{J}')$ with $(t', |\mathcal{J}'|) < (t, |\mathcal{J}|)$ and prove it for $(t, \mathcal{J})$. Assume that $1 \in \mathcal{J}$ (the case $1 \notin \mathcal{J}$ is analogous) and let $\boldsymbol{X}_{-1} = (X_2, \ldots, X_t)$ and $\mathcal{J}_{-1} = \{i - 1 \colon i \in \mathcal{J} \setminus \{1\}\}$. Compute

$$\begin{aligned}
\mathsf{E}_{\boldsymbol{x} \xleftarrow{\mathrm{R}} \boldsymbol{X}} \left[2^{\mathrm{H}_{\boldsymbol{X},\mathcal{J}}(\boldsymbol{x})}\right] &= \sum_{x_1 \in \mathrm{Supp}(X_1)} 2^{-\mathrm{H}_{X_1}(\boldsymbol{x}_1)} \cdot 2^{\mathrm{H}_{X_1}(x_1)} \cdot \mathsf{E}_{\boldsymbol{x} \xleftarrow{\mathrm{R}} \boldsymbol{X}_{-1}|_{X_1 = x_1}} \left[2^{\mathrm{H}_{\boldsymbol{X}_{-1}|_{X_1=x_1},\mathcal{J}_{-1}}(\boldsymbol{x})}\right] \\
&\leq \sum_{x_1 \in \mathrm{Supp}(X_1)} 1 \cdot \left|\mathrm{Supp}((\boldsymbol{X}_{-1})_{\mathcal{J}_{-1}}|_{X_1=x_1})\right| \\
&= \sum_{x_1 \in \mathrm{Supp}(X_1)} \left|\mathrm{Supp}(\boldsymbol{X}_{\mathcal{J}\setminus\{1\}}|_{X_1=x_1})\right| \\
&= \left|\mathrm{Supp}(\boldsymbol{X}_{\mathcal{J}})\right|.
\end{aligned}$$

□

Sub-additivity. The chain rule for Shannon entropy yields that

$$\mathrm{H}(X = (X_1, \ldots, X_t)) = \sum_i \mathrm{H}(X_i | X_1, \ldots, X_{i-1}) \leq \sum_i \mathrm{H}(X_i).$$

The following lemma shows that a variant of the above also holds for sample-entropy.

Lemma 2.11. *For random variables $\boldsymbol{X} = (X_1, \ldots, X_t)$, it holds that*

1. $\mathsf{E}_{\boldsymbol{x} \xleftarrow{R} \boldsymbol{X}} \left[2^{\mathrm{H}_{\boldsymbol{X}}(\boldsymbol{x}) - \sum_t \mathrm{H}_{X_i}(\boldsymbol{x}_i)}\right] \leq 1$, *and*
2. $\mathsf{Pr}_{\boldsymbol{x} \xleftarrow{R} \boldsymbol{X}} \left[H_{\boldsymbol{X}}(\boldsymbol{x}) > \log \frac{1}{\varepsilon} + \sum_{i \in [t]} H_{X_i}(\boldsymbol{x}_i)\right] < \varepsilon$, *for any* $\varepsilon > 0$.

Proof. As in Lemma 2.1, the second part follows from the first by Markov's inequality. For the first part, compute

$$\begin{aligned}
\mathsf{E}_{\boldsymbol{x} \xleftarrow{\mathrm{R}} \boldsymbol{X}} \left[2^{\mathrm{H}_{\boldsymbol{X}}(\boldsymbol{x}) - \sum_t \mathrm{H}_{X_i}(\boldsymbol{x}_i)}\right] &= \sum_{\boldsymbol{x} \in \mathrm{Supp}(\boldsymbol{X})} \mathsf{Pr}\left[\boldsymbol{X} = \boldsymbol{x}\right] \cdot \frac{\prod_{i \in [t]} \mathsf{Pr}\left[X_i = \boldsymbol{x}_i\right]}{\mathsf{Pr}\left[\boldsymbol{X} = \boldsymbol{x}\right]} \\
&= \sum_{\boldsymbol{x} \in \mathrm{Supp}(\boldsymbol{X})} \prod_i \mathsf{Pr}\left[X_i = \boldsymbol{x}_i\right] \\
&\leq 1.
\end{aligned}$$

□

2.4 Hashing

We will use two types of (combinatorial) "hash" functions.

2.4.1 Two-Universal Hashing

Definition 2.12 (Two-universal function family)**.** *A function family $\mathcal{H} = \{h\colon \mathcal{D} \mapsto \mathcal{R}\}$ is* two-universal *if $\forall x \neq x' \in \mathcal{D}$, it holds that $\mathsf{Pr}_{h \xleftarrow{R} \mathcal{H}}[h(x) = h(x')] \leq 1/|\mathcal{R}|$.*

An example of such a function family is the set $\mathcal{H}_{s,t} = \{0,1\}^{s \times t}$ of Boolean matrices, where for $h \in \mathcal{H}_{s,t}$ and $x \in \{0,1\}^s$, we let $h(x) = h \times x$ (i.e., the matrix vector product over GF_2). Another canonical example is $\mathcal{H}_{s,t} = \{0,1\}^s$ defined by $h(x) := h \cdot x$ over $\mathrm{GF}(2^s)$, truncated to its first t bits.

A useful application of two-universal hash functions is to convert a source of high Rényi entropy to a (close to) uniform distribution.

Lemma 2.13 (Leftover hash lemma [24, 23])**.** *Let X be a random variable over $\{0,1\}^n$ with $\mathrm{H}_2(X) \geq k$, let $\mathcal{H} = \{g\colon \{0,1\}^n \mapsto \{0,1\}^m\}$ be two-universal, and let $H \xleftarrow{R} \mathcal{H}$. Then $\mathrm{SD}((H, H(X)), (H, \mathcal{U}_m)) \leq \frac{1}{2} \cdot 2^{(m-k)/2}$.*

2.5 Many-wise Independent Hashing

Definition 2.14 (ℓ-wise independent function family)**.** *A function family $\mathcal{H} = \{h\colon \mathcal{D} \mapsto \mathcal{R}\}$ is* ℓ-wise independent *if for any distinct $x_1, \ldots, x_\ell \in \mathcal{D}$, it holds that $(H(x_1), \ldots, H(x_\ell))$ for $H \xleftarrow{R} \mathcal{H}$ is uniform over $\mathcal{R}^\ell$.*

The canonical example of such an ℓ-wise independent function family is $\mathcal{H}_{s,t,\ell} = (\{0,1\}^s)^\ell$ defined by $(h_0, \ldots, h_{\ell-i})(x) := \sum_{0 \leq i \leq \ell-1} h_i \cdot x^i$ over $\mathrm{GF}(2^s)$, truncated to its first t bits.

It is easy to see that, for $\ell > 1$, an ℓ-wise independent function family is two-universal, but ℓ-wise independent function families, in particular with larger value of ℓ, have stronger guarantees on their output distribution compared with two-universal hashing. We will state, and use, one such guarantee in the construction of statistically hiding commitment schemes presented in Section 6.

2.6 One-way Functions

We recall the standard definition of one-way functions.

Definition 2.15 (one-way functions)**.** *A polynomial-time computable $f\colon \{0,1\}^n \mapsto \{0,1\}^*$ is* one-way *if for every* PPT A

$$\mathsf{Pr}_{y \xleftarrow{R} f(U_{s(n)})}\left[\mathsf{A}(1^n, y) \in f^{-1}(y)\right] = \mathrm{neg}(n). \tag{4}$$

Without loss of generality, cf., [16], it can be assumed that $s(n) = n$ and f is length-preserving (i.e., $|f(x)| = |x|$).

In Section 7 we prove that the existence of inaccessible entropy generators implies that of one-way functions. The reduction uses the seemingly weaker notion of distributional one-way functions. Such a function is easy to compute, but it is hard to compute uniformly random preimages of random images.

Definition 2.16. *A polynomial-time computable* $f\colon \{0,1\}^n \to \{0,1\}^{\ell(n)}$ *is* distributional one-way*, if* $\exists p \in \text{poly}$ *such that*

$$\mathrm{SD}\left((x, f(x))_{x \xleftarrow{R} \{0,1\}^n}, (\mathsf{A}(1^n, f(x)), f(x))_{x \xleftarrow{R} \{0,1\}^n}\right) \geq \frac{1}{p(n)}$$

for any PPTM A *and large enough* n*.*

Clearly, any one-way function is also a distributional one-way function. While the other direction is not necessarily always true, Impagliazzo and Luby [22] showed that the existence of distributional one-way functions implies that of (standard) one-way functions. In particular, they proved that if one-way functions do not exist, then any efficiently computable function has an inverter of the following form.

Definition 2.17 (ξ-inverter)**.** *An algorithm* Inv *is an* ξ-inverter *of* $f\colon \{0,1\}^n \mapsto \{0,1\}^{\ell(n)}$ *if the following holds:*

$$\mathsf{Pr}_{x \xleftarrow{R} \{0,1\}^n; y=f(x)}\left[\mathrm{SD}\left((x')_{x' \leftarrow f^{-1}(y)}, (\mathsf{Inv}(1^n, y))\right) > \xi\right] \leq \xi.$$

Lemma 2.18 ([22, Lemma 1], full details in [21, Theorem 4.2.2])**.** *Assume one-way functions do not exist. Then for any polynomial-time computable function* $f\colon \{0,1\}^n \mapsto \{0,1\}^{\ell(n)}$ *and* $p \in \text{poly}$*, there exists a* PPTM *algorithm* Inv *such that the following holds for infinitely many* n*'s: on security parameter* 1^n*, algorithm* Inv *is a* $1/p(n)$*-inverter of* f_n *(i.e.,* f *is restricted to* $\{0,1\}^n$*).*

Impagliazzo and Luby [22] only gave a proof sketch for the above lemma. The full proof can be found in [21, Theorem 4.2.2] (see more details in [2]).

3 Inaccessible Entropy Generators

In this section we formalize the notion of a block-generator, and the real and accessible entropy of such generators. As discussed in the introduction, these entropies and the gap between them (i.e., the inaccessible entropy of the generator) play a pivotal role in our work.

We begin by informally recalling the definition of a block-generator from the introduction. Let $\mathsf{G}\colon \{0,1\}^n \mapsto (\{0,1\}^*)^m$ be an m-block generator over $\{0,1\}^n$ and let $\mathsf{G}(1^n) = (Y_1, \ldots, Y_m)$ denote the output of G over a uniformly random input. The *real entropy* of G is the (Shannon) entropy in G's output blocks: for each block Y_i we take its entropy conditioned on the previous blocks $Y_{<i} = (Y_1, \ldots, Y_{i-i})$. The *accessible entropy* of an arbitrary, adversarial m-block generator $\widetilde{\mathsf{G}}$, with the same block structure as of G, is the entropy of the block of $\widetilde{\mathsf{G}}$ conditioned not only on the previous blocks but also on the *coins used by* $\widetilde{\mathsf{G}}$ *to generate the previous blocks*. The generator $\widetilde{\mathsf{G}}$ is allowed to flip fresh random coins to generate its next block, and this is indeed the source of entropy in the block (everything else is fixed). We only consider generator $\widetilde{\mathsf{G}}$ whose messages are *always* consistent with G: the support of $\widetilde{\mathsf{G}}$'s messages is contained in that of G.

Moving to the formal definitions, we first define an m-block generator and then define the real and accessible entropy of such a generator. The generators below might get "public parameter" chosen uniformly, and we measure the entropy of the generator conditioned on this parameter.[5]

[5]The public parameter extension is not used for our reduction from one-way functions to statistically hiding commitments presented in this work, but this natural extension was already found useful in the work of Bitansky et al. [3].

Definition 3.1 (Block-generators)**.** *Let n be a security parameter, and let $c = c(n)$, $s = s(n)$ and $m = m(n)$. An m-*block generator *is a function $\mathsf{G}\colon \{0,1\}^c \times \{0,1\}^s \mapsto (\{0,1\}^*)^m$, and it is* efficient *if its running time on input of length $c(n) + s(n)$ is polynomial in n.*

We call parameter n the security parameter*, c the public parameter length, s the* seed length*, m the* number of blocks*, and $\ell(n) = \max_{(z,x)\in\{0,1\}^{c(n)}\times\{0,1\}^{s(n)},i\in[m(n)]} |\mathsf{G}(z,x)_i|$ the* maximal block length *of* G*.*

3.1 Real Entropy

Recall that we are interested in lower bounds on the real entropy of a block-generator. We define two variants of real entropy: real Shannon entropy and real min-entropy. We connect these two notions through the notion of real sample-entropy. In other words, for a fixed m-tuple output of the generator, we ask "how surprising were the blocks output by G in this tuple?" We then get the real Shannon entropy by taking the expectation of this quantity over a random execution and the min-entropy by taking the minimum (up to negligible statistical distance). An alternative approach would be to define the notions through the sum of conditional entropies (as we do in the intuitive description in the introduction). This approach would yield closely related definitions, and in fact exactly the same definition in the case of Shannon entropy (see Lemma 3.4).

Definition 3.2 (Real sample-entropy)**.** *Let G be an m-block generator over $\{0,1\}^c \times \{0,1\}^s$, let $n \in \mathbb{N}$, let Z_n and X_n be uniformly distributed over $\{0,1\}^{c(n)}$ and $\{0,1\}^{s(n)}$ respectively, and let $\mathbf{Y}_n = (Y_1, \ldots, Y_m) = \mathsf{G}(Z_n, X_n)$. For $n \in \mathbb{N}$ and $i \in [m(n)]$, define the* real sample-entropy of $\boldsymbol{y} \in \mathrm{Supp}(Y_1, \ldots, Y_i)$ given $z \in \mathrm{Supp}(Z_n)$ *as*

$$\mathrm{RealH}_{\mathsf{G},n}(\boldsymbol{y}|z) = \sum_{j=1}^{i} \mathrm{H}_{Y_j|Z_n,Y_{<j}}(\boldsymbol{y}_j|z, \boldsymbol{y}_{<j}).$$

We omit the security parameter from the above notation when clear from the context.

Definition 3.3 (Real entropy)**.** *Let G be an m-block generator, and let Z_n and $\mathbf{Y}_n$ be as in Definition 3.2. Generator G has* real entropy at least $k = k(n)$*, if*

$$\mathsf{E}_{(z,\boldsymbol{y})\xleftarrow{R}(Z_n,\mathbf{Y}_n)} \left[\mathrm{RealH}_{\mathsf{G},n}(\boldsymbol{y}|z)\right] \geq k(n)$$

for every $n \in \mathbb{N}$.

The generator G has real min-entropy at least $k(n)$ in its i'th block *for some $i = i(n) \in [m(n)]$, if*

$$\mathsf{Pr}_{(z,\boldsymbol{y})\xleftarrow{R}(Z_n,\mathbf{Y}_n)} \left[\mathrm{H}_{Y_i|Z_n,Y_{<i}}(\boldsymbol{y}_i|z, \boldsymbol{y}_{<i}) < k(n)\right] = \mathrm{neg}(n).$$

We say the above bounds are invariant to the public parameter *if they hold for any fixing of the public parameter Z_n.*[6]

We observe that the real Shannon entropy simply amounts to measuring the standard conditional Shannon entropy of G's output blocks.

Lemma 3.4. *Let G, Z_n and $\mathbf{Y}_n$ be as in Definition 3.2 for some $n \in \mathbb{N}$. Then*

$$\mathsf{E}_{(z,\boldsymbol{y})\xleftarrow{R}(Z_n,\mathbf{Y}_n)} \left[\mathrm{RealH}_{\mathsf{G},n}(\boldsymbol{y}|z)\right] = \mathrm{H}(\mathbf{Y}_n|Z_n).$$

[6] In particular, this is the case when there is no public parameter, i.e., $c = 0$.

Proof. Omit n for clarity. Applying Proposition 2.8(1) to $\boldsymbol{X} = (Z, \mathbf{Y})$ and $\mathcal{J} = \{2, \ldots, m+1\}$ yields that $\mathsf{E}_{(z,\boldsymbol{y}) \stackrel{R}{\leftarrow} (Z,\mathbf{Y})} [\mathrm{RealH}_{\mathsf{G}}(\boldsymbol{y}|z)] = \sum_{i \in [m]} \mathrm{H}(Y_i|Z, Y_{<i})$, where by the chain rule for Shannon entropy , $\sum_{i \in [m]} \mathrm{H}(Y_i|Z, Y_{<i}) = \mathrm{H}(\mathbf{Y}|Z)$. □

3.2 Accessible Entropy

Recall that we are interested in *upper bounds* on the accessible entropy of a block-generator. As in the case of real entropy, we define this notion through the notion of accessible sample-entropy. For a fixed execution of the adversary $\widetilde{\mathsf{G}}$, we ask how surprising were the messages sent by $\widetilde{\mathsf{G}}$, and then get accessible Shannon entropy by taking the expectation of this quantity over a random execution.

Definition 3.5 (Online block-generator)**.** *Let n be a security parameter, and let $c = c(n)$ and $m = m(n)$. An m-block* online *generator is a function $\widetilde{\mathsf{G}}\colon \{0,1\}^c \times (\{0,1\}^v)^m \mapsto (\{0,1\}^*)^m$ for some $v = v(n)$, such that the i'th output block of $\widetilde{\mathsf{G}}$ is a function of (only) its first i input blocks. We denote the* transcript *of $\widetilde{\mathsf{G}}$ over random input by $T_{\widetilde{\mathsf{G}}}(1^n) = (Z, R_1, Y_1, \ldots, R_m, Y_m)$, for $Z \stackrel{R}{\leftarrow} \{0,1\}^c$, $(R_1, \ldots, R_m) \stackrel{R}{\leftarrow} (\{0,1\}^v)^m$ and $(Y_1, \ldots, Y_m) = \widetilde{\mathsf{G}}(Z, R_1, \ldots, R_m)$.*

That is, an online block-generator is a special type of block-generator that tosses fresh random coins before outputting each new block. In the following, we let $\widetilde{\mathsf{G}}(z, r_1, \ldots, r_i)_i$ stand for $\widetilde{\mathsf{G}}(z, r_1, \ldots, r_i, r^*)_i$ for arbitrary $r^* \in (\{0,1\}^v)^{m-i}$ (note that the choice of r^* has no effect on the value of $\widetilde{\mathsf{G}}(z, r_1, \ldots, r_i, r^*)_i$).

Definition 3.6 (Accessible sample-entropy)**.** *Let n be a security parameter, and let $\widetilde{\mathsf{G}}$ be an online $m = m(n)$-block online generator. The* accessible sample-entropy of $\mathbf{t} = (z, r_1, y_1, \ldots, r_m, y_m) \in \mathrm{Supp}(Z, R_1, Y_1 \ldots, R_m, Y_m) = T_{\widetilde{\mathsf{G}}}(1^n)$ *is defined by*

$$\mathrm{AccH}^{\widetilde{\mathsf{G}}}_{n}(\boldsymbol{t}) = \sum_{i=1}^{m} \mathrm{H}_{Y_i|Z,R_{<i}}(y_i|z, r_{<i}).$$

Again, we omit the security parameter from the above notation when clear from the context.

As in the case of real entropy, the expected accessible entropy of a random transcript can be expressed in terms of the standard conditional Shannon entropy.

Lemma 3.7. *Let $\widetilde{\mathsf{G}}$ be an online m-block generator and let $(Z, R_0, R_1, Y_1, \ldots, R_m, Y_m) = T_{\widetilde{\mathsf{G}}}(1^n)$ be its transcript. Then,*

$$\mathsf{E}_{\boldsymbol{t} \stackrel{R}{\leftarrow} T_{\widetilde{\mathsf{G}}}(Z_n, 1^n)} \left[\mathrm{AccH}^{\widetilde{\mathsf{G}}}(\boldsymbol{t})\right] = \sum_{i \in [m]} \mathrm{H}(Y_i|Z, R_{<i}).$$

Proof. Follows similar lines to that of Lemma 3.4. □

The following observation relates the accessible entropy to the accessible sample-entropy of likely events.

Lemma 3.8. *Let $\widetilde{\mathsf{G}}$ be an online m-block generator of maximal block length ℓ and let $T = T_{\widetilde{\mathsf{G}}}(1^n)$. If $\mathsf{Pr}\left[\mathrm{AccH}^{\widetilde{\mathsf{G}}}(T) > k\right] < \varepsilon$, then $\mathsf{E}\left[\mathrm{AccH}^{\widetilde{\mathsf{G}}}(T)\right] \leq (1-\varepsilon)k + \varepsilon(m\ell - \log 1/\varepsilon) < k + \varepsilon m\ell + 1$.*

Proof. Follows by Lemma 2.9 with $\boldsymbol{X} = T_{\widetilde{\mathsf{G}}}(1^n, Z_n) = (Z_n, Y_1, R_1, \ldots, Y_n, R_n)$, for $(Z_n, R_1, Y_1, \ldots, R_n, Y_n) = T_{\widetilde{\mathsf{G}}}(1^n, Z_n)$ and $\mathcal{J} = \{2, 4, 6, \ldots\}$ (i.e., the indices of the output blocks of $\widetilde{\mathsf{G}}$). □

The above definition is only interesting when placing restrictions on the generator's actions with respect to the underlying generator G. (Otherwise, the accessible entropy of $\widetilde{\mathsf{G}}$ can be arbitrarily large by outputting arbitrarily long strings.) In this work, we focus on efficient generators that are consistent with respect to G. That is, the support of their output is contained in that of G.[7]

Definition 3.9 (Consistent generators)**.** *Let G be a block-generator over $\{0,1\}^{c(n)} \times \{0,1\}^{s(n)}$. A block (possibly online) generator G' over $\{0,1\}^{c(n)} \times \{0,1\}^{s'(n)}$ is* G consistent *if, for every $n \in \mathbb{N}$, it holds that $\mathrm{Supp}(\mathsf{G}'(U_{c(n)}, U_{s'(n)})) \subseteq \mathrm{Supp}(\mathsf{G}(U_{c(n)}, U_{s(n)}))$.*

Definition 3.10 (Accessible entropy)**.** *A block-generator G has* accessible entropy at most $k = k(n)$ *if, for every efficient G-consistent, online generator $\widetilde{\mathsf{G}}$ and all large enough n,*

$$\mathsf{E}_{\boldsymbol{t} \xleftarrow{R} T_{\widetilde{\mathsf{G}}}(1^n)} \left[\mathrm{AccH}_{\widetilde{\mathsf{G}}}(\boldsymbol{t}) \right] \leq k.$$

We call a generator whose real entropy is noticeably higher than its accessible entropy an inaccessible entropy generator.

Remark 3.11 (Maximal accessible entropy)**.** *An alternative to the (average) accessible entropy discussed above is to consider the* maximal *accessible sample-entropy of a random execution of the generator (ignoring negligible events). The resulting notion is somewhat harder to work with, but when applicable it typically yields more efficient constructions (time wise and communication wise). Indeed, working with maximal accessible entropy would have yielded a statistically hiding commitment based on one-way functions that is more computation and communication efficient than the one we construct in Section 6. Yet, preferring simplicity over efficiency, in the main body of the paper we chose to work with the simpler notion of (average) accessible entropy, and formally define and prove basic facts on the maximal accessible entropy notion in Appendix A.*

4 Inaccessible Entropy Generator from One-way Functions

In this section we show how to build an inaccessible entropy generator from any one-way function. In particular, we prove the following result:

Construction 4.1. *For $f\colon \{0,1\}^n \mapsto \{0,1\}^n$, define the $(n/\log n)+1$ block-generator G^f over $\{0,1\}^n$,*[8] *by*

$$\mathsf{G}(x) = f(x)_{1,\ldots,\log n}, f(x)_{\log n+1,\ldots,2\log n}, \ldots, f(x)_{n-\log n,\ldots,n}, x$$

Namely, the first $n/\log$ blocks of $\mathsf{G}^f(x)$ are the bits of $f(x)$ portioned into $\log n$-bit blocks, and its final block is x.

Theorem 4.2 (Inaccessible entropy generators from one-way functions)**.** *If $f\colon \{0,1\}^n \mapsto \{0,1\}^n$ is one-way, then the efficient block-generator $\mathsf{G} = \mathsf{G}^f$ defined in Construction 4.1 has accessible entropy $n - \omega(\log n)$.*

[7] In the more complicated notion of accessible entropy considered in [14], the "generator" needs to *prove* that its output blocks are in the support of G, by providing an input of G that would have generated the same blocks. It is also allowed there for a generator to fail to prove the latter with some probability, which requires a measure of accessible entropy that discounts entropy that may come from failing.

[8] We assume for simplicity that $n/\log n \in \mathbb{N}$. Otherwise we "pad" f's output.

Proof. Suppose the theorem does not hold and let $\widetilde{\mathsf{G}}$ be an efficient, G-consistent online block-generator such that

$$\mathsf{E}\left[\mathrm{AccH}_{\widetilde{\mathsf{G}}}(\widetilde{T})\right] \geq n - c \cdot \log n \tag{5}$$

for some $c > 0$, and infinitely many n's. In the following, fix $n \in \mathbb{N}$ for which the above equation holds, and omit it from the notation when its value is clear from the context. Let $m = n/\log n$ and let v be a bound on the number of coins used by $\widetilde{\mathsf{G}}$ in each round. The inverter Inv for f is defined as follows:

Algorithm 4.3 (Inverter Inv for f from the accessible entropy generator $\widetilde{\mathsf{G}}$)**.**

Input: $z \in \{0,1\}^n$

Operation:

1. *For $i = 1$ to m:*
   (a) *Sample $r_i \xleftarrow{R} \{0,1\}^v$ and let $y_i = \widetilde{\mathsf{G}}(r_1, \ldots, r_i)_i$.*
   (b) *If $y_{i\cdot\log n+1,\ldots,(i+1)\cdot\log n} = z_i$, move to next value of i.*
   (c) *Abort after n^3 failed attempts for sampling a good r_i.*
2. *Sample $r_{m+1} \xleftarrow{R} \{0,1\}^v$ and output $\widetilde{\mathsf{G}}(r_1, \ldots, r_{m+1})_{m+1}$.*

Namely, $\mathsf{Inv}(y)$ does the only natural thing that can be done with $\widetilde{\mathsf{G}}$; it tries to make, via rewinding, $\widetilde{\mathsf{G}}$'s first m output blocks equal to y, knowing that if this happens then, since $\widetilde{\mathsf{G}}$ is G-consistent, $\widetilde{\mathsf{G}}$'s $(m+1)$ output block is a preimage of y.

It is clear that Inv runs in polynomial time, so we will finish the proof by showing that it inverts f with high probability. We prove the latter by showing that transcript distribution induced by the (standalone execution) of $\widetilde{\mathsf{G}}(1^n)$ is close in KL-divergence to the execution of $\widetilde{\mathsf{G}}$ embedded (emulated) in $\mathsf{Inv}(f(U_n))$. Since the last output of $\widetilde{\mathsf{G}}$ is always the preimage of the image point defined by its first m output elements, it follows that it is also the case, with high probability, for embedded execution $\widetilde{\mathsf{G}}$, and thus $\mathsf{Inv}(f(U_n))$ finds a preimage with high probability.

Let $\widetilde{T} = (\widetilde{R}_1, \widetilde{Y}_1, \ldots, \widetilde{R}_{m+1}, \widetilde{Y}_{m+1}) = T_{\widetilde{\mathsf{G}}}$, and recall that $\widetilde{T}$ is associated with a random execution of $\widetilde{\mathsf{G}}$ on security parameter n by

- $\widetilde{R}_i$ – the random coins of $\widetilde{\mathsf{G}}$ in the i'th round, and
- $\widetilde{Y}_i$ – $\widetilde{\mathsf{G}}$'s i'th output block.

Let $\widehat{T} = (\widehat{R}_1, \widehat{Y}_1, \ldots, \widehat{R}_{m+1}, \widehat{Y}_{m+1})$ denote the value of $\widetilde{\mathsf{G}}$'s coins and output blocks, set by the end of Step 2 in a random execution of the *unbounded* version of Inv (i.e., Step 1b is removed) on input $f(U_n)$, setting it arbitrary in case the execution of (the unbounded) Inv does not end.[9] The heart of the proof lies in the following claim:

Claim 4.4. $\mathsf{D}\big(\widetilde{T} \,\|\, \widehat{T}\big) \leq c \cdot \log n$.

[9]The unboundedness change is only an intermediate step in the proof that does not significantly change the inversion probability of Inv, as shown below.

We prove Claim 4.4 below but first use it for proving the theorem. Let $w(r_1, y_1, \ldots, r_{m+1}, y_{m+1})$ be the indicator for $y_{m+1} \in f^{-1}(y_{\leq m})$. Clearly,

$$\mathsf{Pr}\left[w(\widetilde{T})\right] = 1 \tag{6}$$

Let $e(r_1, y_1, \ldots, r_{m+1}, y_{m+1})$ be the indicator that for all $i \in [m]$ it holds that $\mathsf{Pr}\left[\widetilde{Y}_i = y_i \mid \widetilde{R}_{<i} = r_{<i}\right] \geq 1/2mn$. Since $\widetilde{Y}_i$ takes at most n values, a straightforward union bound argument yields that

$$\mathsf{Pr}\left[e(\widetilde{T})\right] \geq 1/2 \tag{7}$$

Therefore $\mathsf{Pr}\left[(w \wedge e)(\widetilde{T}) = w(\widetilde{T}) \wedge e(\widetilde{T})\right] \geq 1/2$. By Claim 4.4 and the data-processing property of KL-divergence, $\mathsf{D}\big((w \wedge e)(\widetilde{T}) \,\|\, (w \wedge e)(\widehat{T})\big) \leq c \cdot \log n$. Thus, a simple calculation yields that

$$\mathsf{Pr}\left[(w \wedge e)(\widehat{T})\right] \geq 1/n^{c+1} \tag{8}$$

for large enough n. Since a transcript t with $(w \wedge e)(t) = 1$ is generated by (bounded) Inv with probability at $(1 - m \cdot 2^{-n})$ times the probability it is generated by the unbounded variant of Inv, we deduce that

$$\mathsf{Pr}_{y \xleftarrow{\mathrm{R}} f(U_n)}\left[\mathsf{Inv}(y) \in f^{-1}(y)\right] \geq \mathsf{Pr}\left[(w \wedge e)(\widehat{T})\right] - m \cdot 2^{-n} \geq 1/n^{c+1} - n \cdot 2^{-n}$$

for large enough n, contradicting the one-wayness of f. □

Bounding $\mathsf{D}(\widetilde{T} \,\|\, \widehat{T})$ (proving Claim 4.4).

Proof of Claim 4.4. By definition,

$$\widetilde{R}_i|_{\widetilde{Y}_i = y_i, \widetilde{R}_{<i} = r_{<i}} \equiv \widehat{R}_i|_{\widehat{Y}_i = y_i, \widehat{R}_{<i} = r_{<i}} \tag{9}$$

for every $i \in [m+1]$, $y_i \in \mathrm{Supp}(\widetilde{Y}_i)$ and $r_{<i} \in \mathrm{Supp}(\widetilde{R}_{<i})$, and

$$\widetilde{Y}_{m+1}|_{\widetilde{R}_{\leq m} = r_{<m}} \equiv \widehat{Y}_{m+1}|_{\widehat{R}_{\leq m} = r_{<m}} \tag{10}$$

for every $r_{<m} \in \mathrm{Supp}(\widetilde{R}_{\leq m})$. Compute

$$\begin{aligned}
&\mathsf{D}(\widetilde{T} \,\|\, \widehat{T}) \qquad (11)\\
&= \sum_{i \in \{2,4,\ldots,2m+2\}} \mathsf{E}_{t \xleftarrow{\mathrm{R}} \widetilde{T}_{<i}}\left[\mathsf{D}\big(\widetilde{T}_i|_{\widetilde{T}_{<i}=t} \,\|\, \widehat{T}_i|_{\widehat{T}_{<i}=t}\big)\right] + \sum_{i \in \{1,3,\ldots,2m+1\}} \mathsf{E}_{t \xleftarrow{\mathrm{R}} \widetilde{T}_{<i}}\left[\mathsf{D}\big(\widetilde{T}_i|_{\widetilde{T}_{<i}=t} \,\|\, \widehat{T}_i|_{\widehat{T}_{<i}=t}\big)\right]\\
&= \sum_{i=1}^{m+1} \mathsf{E}_{r \xleftarrow{\mathrm{R}} \widetilde{R}_{<i}}\left[\mathsf{D}\big(\widetilde{Y}_i|_{\widetilde{R}_{<i}=r} \,\|\, \widehat{Y}_i|_{\widehat{R}_{<i}=r}\big)\right] + \sum_{i=1}^{m+1} \mathsf{E}_{(r,y) \xleftarrow{\mathrm{R}} (\widetilde{R}_{<i}, \widetilde{Y}_i)}\left[\mathsf{D}\big(\widetilde{R}_i|_{\widetilde{R}_{<i}=r, \widetilde{Y}_i=y} \,\|\, \widehat{R}_i|_{\widehat{R}_{<i}=r}, \widehat{Y}_i = y\big)\right]\\
&= \sum_{i=1}^{m} \mathsf{E}_{r \xleftarrow{\mathrm{R}} \widetilde{R}_{<i}}\left[\mathsf{D}\big(\widetilde{Y}_i|_{\widetilde{R}_{<i}=r} \,\|\, \widehat{Y}_i|_{\widehat{R}_{<i}=r}\big)\right] + 0 + 0.
\end{aligned}$$

The first equality holds by chain-rule of KL-divergence, and since KL-divergence is invariant to permutation. The last equality holds by Equations (9) and (10). It follows that

$$\begin{aligned}
\mathsf{D}\big(\widetilde{T} \,\|\, \widehat{T}\big) &= \sum_{i=1}^{m} -\mathrm{H}(\widetilde{Y}_i \mid \widetilde{R}_{<i}) - \sum_{i=1}^{m} \mathsf{E}_{r_{<i} \xleftarrow{\mathrm{R}} \widetilde{R}_{<i}} \left[\log\left(\mathsf{Pr}\left[\widehat{Y}_i = y_i \mid \widehat{R}_{<i} = r_{<i}\right]\right)\right] \qquad (12)\\
&= \sum_{i=1}^{m} -\mathrm{H}(\widetilde{Y}_i \mid \widetilde{R}_{<i}) - \sum_{i=1}^{m} \mathsf{E}_{y_{<i} \xleftarrow{\mathrm{R}} \widetilde{Y}_{<i}} \left[\log\left(\mathsf{Pr}\left[\widehat{Y}_i = y_i \mid \widehat{Y}_{<i} = y_{<i}\right]\right)\right]\\
&= \sum_{i=1}^{m} -\mathrm{H}(\widetilde{Y}_i \mid \widetilde{R}_{<i}) - \mathsf{E}_{y_{\le m} \xleftarrow{\mathrm{R}} \widetilde{Y}_{\le m}} \left[\sum_{i=1}^{m} \log\left(\mathsf{Pr}\left[\widehat{Y}_i = y_i \mid \widehat{Y}_{<i} = y_{<i}\right]\right)\right]\\
&= \sum_{i=1}^{m} -\mathrm{H}(\widetilde{Y}_i \mid \widetilde{R}_{<i}) - \mathsf{E}_{y_{\le m} \xleftarrow{\mathrm{R}} \widetilde{Y}_{\le m}} \left[\log\left(\mathsf{Pr}\left[\widehat{Y}_{\le m} = y_{\le m}\right]\right)\right]\\
&= \mathrm{H}(\widetilde{Y}_{m+1} \mid \widetilde{R}_{\le m}) - \mathsf{E}\left[\mathrm{AccH}_{\widetilde{\mathsf{G}}}(\widetilde{T})\right] - \mathsf{E}_{y_{\le m} \xleftarrow{\mathrm{R}} \widetilde{Y}_{\le m}} \left[\log\left(\mathsf{Pr}\left[\widehat{Y}_{\le m} = y_{\le m}\right]\right)\right].
\end{aligned}$$

The first inequality holds by Equation (11) and definition of KL-divergence. The second equality holds since $\widehat{Y}_i$ and $\widehat{R}_{<i}$ are independent conditioned on $\widehat{Y}_{<i}$. The last equality is by the definition of $\mathrm{AccH}_{\widetilde{\mathsf{G}}}$. Since $\mathrm{H}(\widetilde{Y}_{m+1} \mid \widetilde{R}_{\le m}) = \mathsf{E}_{y_{\le m} \xleftarrow{\mathrm{R}} \widetilde{Y}_{\le m}} \left[\log\left|f^{-1}(y_{\le m})\right|\right]$ and $\mathsf{Pr}\left[\widehat{Y}_{\le m} = y_{\le m}\right] = \left|f^{-1}(y_{\le m})\right|/2^n$, Equation (12) yields that

$$\mathsf{D}\big(\widetilde{T} \,\|\, \widehat{T}\big) \le -\mathsf{E}\left[\mathrm{AccH}_{\widetilde{\mathsf{G}}}(\widetilde{T})\right] + n \le c \cdot \log n.$$

The second inequality is by the assumption on $\widetilde{\mathsf{G}}$ accessible entropy (Equation (5)). □

5 Manipulating Real and Accessible Entropy

In this section, we develop tools to manipulate the real and accessible entropy of a block-generator. These tools are later used in Section 6 as a first step toward using the one-way function inaccessible entropy generator constructed in Section 4 to construct statistically hiding commitments. The tools considered below are rather standard "entropy manipulations": entropy equalization (i.e., picking a random variable at random from a set of random variables to get a new random variable whose entropy is the average entropy) and direct product, and their effect on the real entropy of random variables is rather clear. Fortunately, these manipulations have the same effect also on the *accessible entropy* of a block-generator.

5.1 Entropy Equalization via Truncated Sequential Repetition

This tool concatenates several independent executions of an m-block generator, and then truncates, at random, some of the first and final output blocks of the concatenated string. Assuming that the (overall) real entropy of the original generator is at least k_{REAL}, then the real entropy of each block of the resulting generator is at least k_{REAL}/m. This per-block knowledge of the real entropy is very handy when considering applications of inaccessible entropy generators, and in particular for constructions of statistically hiding commitments (see Section 6). The price of this manipulation is that we "give away" some real entropy (as we do not output all blocks), while we cannot claim

that the same happens to the accessible entropy. Hence, the additive gap between the real and accessible entropy of the resulting generator gets smaller. Yet, if we do enough repetition, this loss is insignificant.

Definition 5.1. *For security parameter n, let $m = m(n)$, let $c = c(n)$, $s = s(n)$, $w = w(n)$, and let $c' = c'(n) = w(n) \cdot c(n)$ and $s' = s'(n) = \log(m(n)) + w(n) \cdot s(n)$. Given an m-block generator G over $\{0,1\}^c \times \{0,1\}^s$, define the $((w-1) \cdot m)$-block generator $\mathsf{G}^{[w]}$ over $(\{0,1\}^c)^w \times ([m] \times (\{0,1\}^s)^w)$ as follows: on input $(z_1, \ldots, z_w, j, (x_1, \ldots, x_w))$, it sets $\boldsymbol{y} = (y_1, \ldots, y_{wm}) = (\mathsf{G}(z_1, x_1), \ldots, \mathsf{G}(z_w, x_w))$, and outputs $((j, y_j), y_{j+1}, \ldots, y_{(w-1)m+j-1})$.*

That is, $\mathsf{G}^{[w]}$ truncates the first $j-1$ and last $m+1-j$ blocks of $\boldsymbol{y}$. It then outputs the remaining $(w-1) \cdot m$ blocks of $\boldsymbol{y}$ one by one, while appending j, indicating the truncation location, to the first block.

Lemma 5.2. *For security parameter n, let $m = m(n)$ be a power of 2, let $c = c(n)$, $s = s(n)$, let G be an efficient m-block generator over $\{0,1\}^c \times \{0,1\}^s$, and let $w = w(n)$ be a polynomially computable and bounded integer function. Then $\mathsf{G}^{[w]}$ defined according to Definition 5.1 is an efficient[10] $((w-1) \cdot m)$-block generator satisfying the following properties:*

Real entropy: *If G has real entropy at least $k_{\text{REAL}} = k_{\text{REAL}}(n)$, then each block of $\mathsf{G}^{[w]}$ has real entropy at least k_{REAL}/m. Furthermore, if the bound on the real entropy G is invariant to the public parameter, then this is also the case for the above bound on $\mathsf{G}^{[w]}$.*

Accessible entropy: *If G has accessible entropy at most $k_{\text{ACC}} = k_{\text{ACC}}(n)$, then $\mathsf{G}^{[w]}$ has accessible entropy at most*

$$k'_{\text{ACC}} := (w-2) \cdot k_{\text{ACC}} + 2 \cdot \mathrm{H}_0(\mathsf{G}(U_c, U_s)) + \log(m).$$

That is, each of the $(w-2)$ non-truncated executions of G embedded in $\mathsf{G}^{[w]}$ contributes its accessible entropy to the overall accessible entropy of $\mathsf{G}^{[w]}$. In addition, we pay the max-entropy of the two truncated executions of G embedded in $\mathsf{G}^{[w]}$ and the max-entropy of the index j.

Proof. To avoid notational clutter let $\mathbb{G} = \mathsf{G}^{[w]}$.

Real entropy. Fix $n \in \mathbb{N}$ and omit it from the notation when clear from the context. Let $\tilde{m} = (w-1)m$, let $\mathbf{Z} = (Z_1, \ldots, Z_w) \xleftarrow{R} (\{0,1\}^c)^w$ and $\widetilde{Y} = \mathsf{G}^{[w]}(\mathbf{Z}, U_{s'} = (J, X_1, \ldots, X_w))$. Let $Y^{(w)} = (\mathsf{G}(Z_1, X_1), \ldots, \mathsf{G}(Z_w, X_w))$, and finally for $i \in [wm]$, let $\widehat{Y}_i^{(w)} = (J, Y_i^{(w)})$ if $J = i$, and $\widehat{Y}_i^{(w)} = Y_i^{(w)}$ otherwise.

Fix $i \in [\tilde{m}]$. By the chain rule for Shannon entropy, it holds that

$$\begin{aligned} \mathrm{H}(\widetilde{Y}_i \mid Z, \widetilde{Y}_{<i}) &= \mathrm{H}(\widehat{Y}^{(w)}_{i+J-1} \mid \mathbf{Z}, \widehat{Y}^{(w)}_{J,\ldots,i+J-2}) \\ &\geq \mathrm{H}(\widehat{Y}^{(w)}_{i+J-1} \mid \mathbf{Z}, Y^{(w)}_{J,\ldots,i+J-2}, J). \\ &= \mathrm{H}(Y^{(w)}_{i+J-1} \mid \mathbf{Z}, Y^{(w)}_{J,\ldots,i+J-2}, J). \end{aligned} \tag{13}$$

[10] Since m is a power of 2, standard techniques can be used to change the input domain of $\mathsf{G}^{[w]}$ to $\{0,1\}^{s'}$ for some polynomial-bounded and polynomial-time computable s', to make it an efficient block-generator according to Definition 3.1.

Let $(Y_1,\ldots,Y_m)=\mathsf{G}(Z=U_c,U_s)$, and let $m \bmod m$ be m (rather than 0). It follows that

$$\begin{aligned}
\mathrm{H}(Y^{(w)}_{i+J-1} \mid Z, Y^{(w)}_{<i+J-1}, J) &= \mathsf{E}_{j\xleftarrow{R} J}\left[\mathrm{H}(Y^{(w)}_{i+j-1} \mid \mathbf{Z}, Y^{(w)}_{<i+j-1})\right] \\
&= \mathsf{E}_{j\xleftarrow{R} J}\left[\mathrm{H}(Y_{i+j-1 \bmod m} \mid Z, Y_{<i+J-1 \bmod m})\right] \\
&= \mathsf{E}_{i'\xleftarrow{R}[m]}[\mathrm{H}(Y_{i'} \mid Z, Y_{<i'})] \\
&= \frac{1}{m}\cdot\sum_{i'\in[m]} \mathrm{H}(Y_{i'} \mid Z, Y_{<i'}) \\
&\geq k_{\mathrm{REAL}}/m,
\end{aligned}$$

yielding that $\mathrm{H}(\widetilde{Y}_i \mid \mathbf{Z}, \widetilde{Y}_{<i}) \geq k_{\mathrm{REAL}}/m$. The second equality follows from the fact that, for any $t\in[wm]$, $(Z_{\lceil t/m\rceil}, Y^{(w)}_{t'=\lfloor t/m\rfloor\cdot m+1},\ldots,Y^{(w)}_t)$ is independent of $(\mathbf{Z}_{<\lceil t/m\rceil}, \mathbf{Z}_{>\lceil t/m\rceil}, Y^{(w)}_{<t'})$, and has the same distribution as $(Z,Y_1,\ldots,Y_{t \bmod m})$. The third equality holds since $(i+J-1 \bmod m)$ is uniformly distributed in $[m]$.

It readily follows from the above proof that if the bound on the real entropy G is invariant to the public parameter, then the above k_{REAL}/m bound on the real entropy of $\mathsf{G}^{[w]}$ is also invariant to the public parameter.

Accessible entropy. Let $\widetilde{\mathbb{G}}$ be an efficient $\mathbb{G}$ $(=\mathsf{G}^{[w]})$-consistent generator, and assume

$$k_{\widetilde{\mathbb{G}}} = \mathsf{E}_{\mathbf{t}\xleftarrow{R}\widetilde{\mathbb{T}}}\left[\mathrm{AccH}^{\widetilde{\mathbb{G}}}(\mathbf{t})\right] > k'_{\mathrm{ACC}} \tag{14}$$

for $\widetilde{\mathbb{T}} = T_{\widetilde{\mathbb{G}}}(1^n)$. We show that Equation (14) yields that a random sub-transcript of $\widetilde{\mathbb{T}}$ contributes more than k_{ACC} bits of accessible entropy, and use it to construct a cheating generator for G whose accessible entropy is greater than k_{ACC}, in contradiction to the assumed accessible entropy of G.

Let $(\mathbf{Z},R_1,Y_1,\ldots,R_{\tilde{m}},Y_{\tilde{m}})=\widetilde{\mathbb{T}}$ and let J be the first part of Y_1. Since $\widetilde{\mathbb{G}}$ is $\mathbb{G}$-consistent, Y_1 is of the form $(J,\cdot)$. Fix $j\in[m]$, let $(\mathbf{Z},R^j_1,Y^j_1,\ldots,R^j_{\tilde{m}},Y^j_{\tilde{m}})=\widetilde{\mathbb{T}}^j=\widetilde{\mathbb{T}}|_{J=j}$ and let $\mathcal{I}=\mathcal{I}(j)$ be the indices of the output blocks coming from the truncated executions of G in $\mathbb{G}$ (i.e., $\mathcal{I}=\{1,\ldots,m+1-j\}\cup\{\tilde{m}+2-j,\ldots,\tilde{m}\}$). It is easy to see that these blocks do not contribute more entropy than twice the max-entropy of G. Indeed Proposition 2.8, letting $\boldsymbol{X}=(\mathbf{Z},Y^j_1,R^j_1,\ldots,Y^j_{\tilde{m}},R^j_{\tilde{m}})$ and $\mathcal{J}$ being the indices of the blocks of $\mathcal{I}$ in $\boldsymbol{X}$, yields that

$$\mathsf{E}_{(\boldsymbol{z},r_1,y_1,\ldots,r_{\tilde{m}},y_{\tilde{m}})\xleftarrow{R}\widetilde{\mathbb{T}}^j}\left[\sum_{i\in\mathcal{I}}\mathrm{H}_{Y^j_i|\mathbf{Z},R^j_{<i}}(y_i|\boldsymbol{z},r_{<i})\right] \leq 2\cdot\mathrm{H}_0(\mathsf{G}(U_c,U_s)) \tag{15}$$

Consider $(\boldsymbol{z},r_1,y_1=(j,\cdot),\ldots,r_{\tilde{m}},y_{\tilde{m}})\in\mathrm{Supp}(\widetilde{\mathbb{T}})$. Since J is determined by (Z,R_1), for $i>1$ it holds that $\mathrm{H}_{Y_i|\mathbf{Z},R_{<i}}(y_i|\boldsymbol{z},r_{<i})=\mathrm{H}_{Y^j_i|\mathbf{Z},R^j_{<i}}(y_i|\boldsymbol{z},r_{<i})$, whereas for $i=1$, $\mathrm{H}_{Y_1|\mathbf{Z}}(y_1|\boldsymbol{z})=\mathrm{H}_{J|\mathbf{Z}}(j|\boldsymbol{z})+\mathrm{H}_{Y^j_1|\mathbf{Z}}(y_1|\boldsymbol{z})$, where $\mathsf{E}_{(\boldsymbol{z},j)\xleftarrow{R}(Z,J)}\left[\mathrm{H}_{J|\mathbf{Z}}(j|\boldsymbol{z})\right]\leq\log m$. These observations, and Equations (14) and (15), yield that

$$\begin{aligned}
\mathsf{E}_{(\boldsymbol{z},r_1,y_1=(j,\cdot),\ldots,r_{\tilde{m}},y_{\tilde{m}})\xleftarrow{R}\widetilde{\mathbb{T}}}\left[\sum_{i\in[\tilde{m}]\setminus\mathcal{I}(j)}\mathrm{H}_{Y_i|\mathbf{Z},R_{<i}}(y_i|\boldsymbol{z},r_{<i})\right] &\geq k_{\widetilde{\mathbb{G}}} - 2\cdot\mathrm{H}_0(\mathsf{G}(U_c,U_s)) - \log m \\
&> (w-2)\cdot k_{\mathrm{ACC}},
\end{aligned} \tag{16}$$

and we conclude that

$$\mathsf{E}_{(\boldsymbol{z},r_1,y_1=(j,\cdot),\ldots,r_{\tilde{m}},y_{\tilde{m}})\xleftarrow{R}\widetilde{\mathbb{T}};v\xleftarrow{R}\{2,\ldots,w-1\}}\left[\sum_{i\in\{f=(v-1)m+2-j,\ldots,f+m-1\}}\mathrm{H}_{Y_i|\mathbf{Z},R_{<i}}(y_i|\boldsymbol{z},r_{<i})\right]>k_{\mathrm{ACC}} \quad (17)$$

Consider the following efficient G-consistent generator.

Algorithm 5.3 (Generator $\widetilde{\mathsf{G}}$)**.**

Input: public parameter $z\in\{0,1\}^c$.

Operation:

1. *Sample* $v\xleftarrow{R}\{2,\ldots,w-1\}$. *Let* $\boldsymbol{z}=(z_1,\ldots,z_w)$ *for* $z_v=z$, *and* z_i, *for* $i\neq v$, *sampled uniformly in* $\{0,1\}^c$.

   We will refer to the part of $\boldsymbol{z}$ *sampled by the generator as* $\boldsymbol{z}_{-v}$, *and (abusing notation) assume* $\boldsymbol{z}=(z,\boldsymbol{z}_{-v})$.

2. *Start a random execution of* $\widetilde{\mathbb{G}}(\boldsymbol{z})$. *After* $\widetilde{\mathbb{G}}$ *locally outputs its first block* $(j,\cdot)$, *continue the execution of* $\widetilde{\mathbb{G}}$ *while outputting, block by block, the output blocks of* $\widetilde{\mathbb{G}}$ *indexed by* $\{f=(v-1)m+2-j,\ldots,f+m-1\}$.

It is clear that $\widetilde{\mathsf{G}}$ is indeed an efficient G-consistent generator. We will show that the accessible entropy of $\widetilde{\mathsf{G}}$ violates the assumed bound on the accessible entropy of G.

Let $(Z,\widetilde{R}_1=(V,\mathbf{Z}_{-V},F,R'_{<F},R'_F),\widetilde{Y}_1,\ldots,\widetilde{R}_m,\widetilde{Y}_m)=T_{\widetilde{\mathsf{G}}}$ be the transcript of $\widetilde{\mathsf{G}}(Z)$, for $Z\xleftarrow{R}\{0,1\}^c$, $(V,\mathbf{Z}_{-V},F)$ being the value of $(v,\boldsymbol{z}_{-v},f)$ sampled in the first step of $\widetilde{\mathsf{G}}$, and $R'_{<F}$ and R'_F being the randomness used by the emulated execution of $\widetilde{\mathbb{G}}(Z,\mathbf{Z}_{-V})$ done in the second step of $\widetilde{\mathsf{G}}$, for outputting its first $(F-1)$ blocks and the F'th block, respectively. Let $(z,\widetilde{r}_1=(v,\boldsymbol{z}_{-v},f,r'_{<f},r'_f),y_1,\ldots,\widetilde{r}_m,y_m)\in\mathrm{Supp}(T_{\widetilde{\mathsf{G}}})$. It is easy to verify that

$$\mathrm{H}_{\widetilde{Y}_1|Z,V,Z_{-V},F,R'_{<F}}(y_1\mid z,v,\boldsymbol{z}_{-v},f,r'_{<f})=\mathrm{H}_{Y_f|\mathbf{Z},R_{<f}}(y_1\mid(z,\boldsymbol{z}_{-f}),r_{<f}) \quad (18)$$

and that for $i>1$:

$$\mathrm{H}_{\widetilde{Y}_i|Z,\widetilde{R}_{<i}}(y_i\mid z,\widetilde{r}_{<i})=\mathrm{H}_{Y_{f+i}|\mathbf{Z},R_{<f+i}}(y_i\mid(z,\boldsymbol{z}_{-f}),(r'_{<f},r'_f,\widetilde{r}_2,\ldots,\widetilde{r}_{i-1})) \quad (19)$$

It follows that

$$\begin{aligned}
&\mathsf{E}_{(\boldsymbol{z},r_1,y_1=(j,\cdot),\ldots,r_{\tilde{m}},y_{\tilde{m}})\xleftarrow{R}\widetilde{\mathbb{I}};v\xleftarrow{R}\{2,\ldots,w-1\}}\left[\sum_{i\in\{f=(v-1)m+2-j,\ldots,f+m-1\}}\mathrm{H}_{Y_i|\mathbf{Z},R_{<i}}(y_i|\boldsymbol{z},r_{<i})\right]\\
&=\mathsf{E}_{(z,\widetilde{r}_1=(v,\boldsymbol{z}_{-v},f,r'_{<f},r'_f),y_1,\ldots,\widetilde{r}_m,y_m)\xleftarrow{R}T_{\widetilde{\mathsf{G}}}}\left[\mathrm{H}_{\widetilde{Y}_1|Z,V,Z_{-V},F,R'_{<F}}(y_i\mid z,v,\boldsymbol{z}_{-v},f,r'_{<f})+\sum_{i=2}^{m}\mathrm{H}_{\widetilde{Y}_i|Z,\widetilde{R}_{<i}}(y_i\mid z,\widetilde{r}_{<i})\right]\\
&=\mathrm{H}(\widetilde{Y}_1\mid Z,V,Z_{-V},F,R'_{<F})+\sum_{i=2}^{m}\mathrm{H}(\widetilde{Y}_i\mid Z,\widetilde{R}_{<i})\\
&\leq\mathrm{H}(\widetilde{Y}_1\mid Z)+\sum_{i=2}^{m}\mathrm{H}(\widetilde{Y}_i\mid Z,\widetilde{R}_{<i})\\
&=\mathsf{E}_{\mathbf{t}\xleftarrow{R}T_{\widetilde{\mathsf{G}}}}\left[\mathrm{AccH}_{\widetilde{\mathsf{G}}}(\mathbf{t})\right].
\end{aligned}$$

The inequality is by the chain rule of Shannon entropy and the last equality by Lemma 3.7. Thus, by Equation (17), the accessible entropy of $\widetilde{\mathsf{G}}$ is greater than k_{ACC}, in contradiction to the assumed accessible entropy of G. □

5.2 Gap Amplification Via Direct Product

This manipulation simply takes the direct product of a generator. The effect of this manipulation is two-fold. The first effect is that the overall real entropy of a v-fold direct product repetition of a generator G is v times the real entropy of G. Hence, if G's real entropy is larger than its accessible entropy, this gap gets multiplied by v when we perform direct product. The second effect of such repetition is that per-block real entropy is turned into per-block *min-entropy*. The price of this manipulation is a slight decrease in the per block min-entropy of the resulting generator, compared to the sum of the per block real entropies of the independent copies of the generators used to generate it. (This loss is due to the move from Shannon entropy to min-entropy, rather than from the direct product itself.) But when performing sufficient repetitions, this loss can be ignored.

Definition 5.4. *Let $m=m(n)$, $c=c(n)$ and $s=s(n)$, and $v=v(n)$. Given an m-block generator G over $\{0,1\}^c\times\{0,1\}^s$, define the m-block generator $G^{\langle v\rangle}$ over $(\{0,1\}^c)^v\times(\{0,1\}^s)^v$ as follows: on input $(\boldsymbol{z},\boldsymbol{x})$, the i'th block of $G^{\langle v\rangle}$ is $(\mathsf{G}(\boldsymbol{z}_1,\boldsymbol{x}_1)_i,\ldots,\mathsf{G}(\boldsymbol{z}_v,\boldsymbol{x}_v)_i)$.*

Lemma 5.5. *For security parameter n, let $m=m(n)$, let $v=v(n)$ be polynomial-time computable and bounded integer functions, and let G be an efficient[11] m-block generator. Then $G^{\langle v\rangle}$, defined according to Definition 5.4, is an efficient m-block generator that satisfies the following properties:*

Real entropy: *If the i'th block of G has real min-entropy at least $k_{\mathrm{REAL}}=k_{\mathrm{REAL}}(n)$, then the i'th block of $G^{\langle v\rangle}$ has real min-entropy at least $k'_{\mathrm{REAL}}(n)=v\cdot k_{\mathrm{REAL}}-O((\log n+\ell)\cdot\log n\cdot\sqrt{v})$, for $\ell=\ell(n)$ being the length of the i'th block. If the bound on the real entropy G is invariant to the public parameter, then this is also the case for the above bound on $G^{\langle v\rangle}$.*

[11] Again, standard techniques can be used to change the input domain of G to $\{0,1\}^{s'(n)}$ for some polynomial-bounded and polynomial-time computable s', to make it an efficient block-generator according to Definition 3.1.

Accessible entropy: *If* G *has accessible entropy at most* $k_{\text{ACC}} = k_{\text{ACC}}(n)$*, then* $G^{\langle v \rangle}$ *has accessible entropy at most* $v \cdot k_{\text{ACC}}$*.*

Proof. The bound on real entropy follows readily from Lemma 2.6 by taking $\varepsilon = 2^{-\log^2 n}$, and noting that the support size of each block of G is at most $\ell \cdot 2^{\ell}$. Therefore, we focus on establishing the bound on accessible entropy. Let $\mathbb{G} = G^{\langle v \rangle}$, let $\widetilde{\mathbb{G}}$ be an efficient, nonuniform, $\mathbb{G}$-consistent generator, and assume

$$k_{\widetilde{\mathbb{G}}} = \mathsf{E}_{\mathbf{t} \xleftarrow{R} \widetilde{\mathbb{T}}} \left[\text{AccH}^{\widetilde{\mathbb{G}}}(\mathbf{t}) \right] > k'_{\text{ACC}} = v \cdot k_{\text{ACC}} \tag{20}$$

for $(\mathbf{Z}, \boldsymbol{R}_1, \mathbf{Y}_1, \ldots, \boldsymbol{R}_m, \mathbf{Y}_m) = \widetilde{\mathbb{T}} = T_{\widetilde{\mathbb{G}}}(1^n)$. We show that Equation (20) yields that a random "column" of $\widetilde{\mathbb{T}}$ contributes more than k_{ACC} bits of accessible entropy, and use it to construct a cheating generator for G whose accessible entropy is greater than k_{ACC}, in contradiction to the assumed accessible entropy of G.

Let $(\mathbf{Z}, \boldsymbol{R}_1, \mathbf{Y}_1, \ldots, \boldsymbol{R}_m, \mathbf{Y}_m) = \widetilde{\mathbb{T}}$. Since $\widetilde{\mathbb{G}}$ is $\mathbb{G}$-consistent, each Y_i is of the form $(Y_{i,1}, \ldots, Y_{i,v})$. It follows that

$$\begin{aligned} k_{\widetilde{\mathbb{G}}} &= \sum_{i=1}^{m} \text{H}(\mathbf{Y}_i \mid \mathbf{Z}, \boldsymbol{R}_{\leq i}) \\ &= \sum_{i \in [m]} \sum_{j \in [v]} \text{H}(Y_{i,j} \mid \mathbf{Z}, \boldsymbol{R}_{<i}, \mathbf{Y}_{i,1}, \ldots, \mathbf{Y}_{i,j-1}) \\ &\leq \sum_{i \in [m]} \sum_{j \in [v]} \text{H}(Y_{i,j} \mid \mathbf{Z}, \boldsymbol{R}_{<i}, Y_{<j}) \\ &= \sum_{j \in [v]} k_j \end{aligned} \tag{21}$$

for $k_j = \sum_{i \in [m]} \text{H}(Y_{i,j} \mid \mathbf{Z}, \boldsymbol{R}_{<i})$. The first equality is by Lemma 3.7, and the inequality is by the chain rule of Shannon entropy.

Consider the following efficient G-consistent generator.

Algorithm 5.6 (Generator $\widetilde{\mathsf{G}}$)**.**

Input: public parameter $z \in \{0,1\}^c$*.*

Operation:

1. *Sample* $j \xleftarrow{R} [v]$*. Let* $\boldsymbol{z} = (z_1, \ldots, z_w)$ *for* $z_v = z$*, and* z_i*, for* $i \neq v$*, sampled uniformly in* $\{0,1\}^c$*.*

   We will refer to the part of $\boldsymbol{z}$ *sampled by the generator as* $\boldsymbol{z}_{-v}$*, and (abusing notation) assume* $\boldsymbol{z} = (z, \boldsymbol{z}_{-v})$*.*

2. *Start a random execution of* $\widetilde{\mathbb{G}}(\boldsymbol{z})$ *and output the* j*'th entry of each output block.*

It is clear that $\widetilde{\mathsf{G}}$ is indeed an efficient G-consistent generator. We will show that the accessible entropy of $\widetilde{\mathsf{G}}$ violates the assumed bound on the accessible entropy of G.

Let $(Z, \widetilde{R}_1 = (J, \mathbf{Z}_{-J}, R'_1), \widetilde{Y}_1, \ldots, \widetilde{R}_m, \widetilde{Y}_m) = T_{\widetilde{\mathsf{G}}}$, for $(J, \mathbf{Z}_{-J})$ be the value of $(j, \boldsymbol{z}_{-f})$ sampled in the first step of $\widetilde{\mathsf{G}}$ with R'_1 being the randomness used by the emulated $\widetilde{\mathbb{G}}$ to generate its first output bock. Let $(z, \widetilde{r}_1 = (j, \boldsymbol{z}_{-j}, r'_1), y_1, \ldots, \widetilde{r}_m, y_m) \in \mathrm{Supp}(T_{\widetilde{\mathsf{G}}})$. It is easy to verify that

$$\mathrm{H}_{\widetilde{Y}_1|Z,J,\mathbf{Z}_{-J}}(y_i \mid z, j, \boldsymbol{z}_{-j}) = \mathrm{H}_{Y_{1,j}|\mathbf{Z}}(y_1 \mid (z, \boldsymbol{z}_{-f})) \tag{22}$$

and that for $i > 1$:

$$\mathrm{H}_{\widetilde{Y}_i|Z,\widetilde{R}_{<i}}(y_i \mid z, \widetilde{r}_{<i}) = \mathrm{H}_{Y_{i,j}|\mathbf{Z},R_{<i}}(y_i \mid (z, \boldsymbol{z}_{-f}), (r'_1, \widetilde{r}_2, \ldots, \widetilde{r}_{i-1})) \tag{23}$$

It follows that

$$\begin{aligned}
&\mathsf{E}_{j \xleftarrow{R} [v]}[k_j] \\
&\mathsf{E}_{(\boldsymbol{z}, r_1, y_1, \ldots, r_m, y_m) \xleftarrow{R} \widetilde{\mathbb{I}}; j \xleftarrow{R} [v]} \left[\sum_{i \in [m]} \mathrm{H}_{Y_{i,j}|\mathbf{Z},\widetilde{R}_{<i}}(y_i | \boldsymbol{z}, r_{<i}) \right] \\
&= \mathsf{E}_{(z, \widetilde{r}_1 = (j, \boldsymbol{z}_{-j}, r'_1), y_1, \ldots, \widetilde{r}_m, y_m) \xleftarrow{R} T_{\widetilde{\mathsf{G}}}} \left[\mathrm{H}_{\widetilde{Y}_1|Z,J,Z_{-J}}(y_i \mid z, j, \boldsymbol{z}_{-j}) + \sum_{i=2}^{m} \mathrm{H}_{\widetilde{Y}_i|Z,\widetilde{R}_{<i}}(y_i \mid z, \widetilde{r}_{<i}) \right] \\
&= \mathrm{H}(\widetilde{Y}_1 \mid Z, J, Z_{-J}) + \sum_{i=2}^{m} \mathrm{H}(\widetilde{Y}_i \mid Z, \widetilde{R}_{<i}) \\
&\leq \mathrm{H}(\widetilde{Y}_1 \mid Z) + \sum_{i=2}^{m} \mathrm{H}(\widetilde{Y}_i \mid Z, \widetilde{R}_{<i}) \\
&= \mathsf{E}_{\mathbf{t} \xleftarrow{R} T_{\widetilde{\mathsf{G}}}} \left[\mathrm{AccH}_{\widetilde{\mathsf{G}}}(\mathbf{t}) \right].
\end{aligned}$$

The inequality is by the chain rule of Shannon entropy and the last equality by Lemma 3.7. Thus, by Equation (21), the accessible entropy of $\widetilde{\mathsf{G}}$ is greater than k_{ACC}, in contradiction to the assumed accessible entropy of G. □

6 Entropy Gap to Commitment

In this section we construct statistically hiding commitments from inaccessible entropy generators. Combined with the main result of Section 4, the above yields a construction of statistically hiding commitments from one-way functions, reproving the result of [13].

We start by recalling the definition of statistically hiding commitment schemes.

Statistically hiding commitment schemes. A *commitment scheme* is the cryptographic analogue of a safe. It is a two-party protocol between a *sender* S and a *receiver* R that consists of two stages. The *commit stage* corresponds to putting an object in a safe and locking it; the sender "commits" to a private message m. The *reveal stage* corresponds to unlocking and opening the safe; the sender "reveals" the message m and "proves" that it was the value committed to in the commit stage (without loss of generality by revealing coin tosses consistent with m and the transcript of the commit stage).

Definition 6.1. *A* (bit) commitment scheme[12] *is an efficient two-party protocol* $\mathsf{Com} = (\mathsf{S}, \mathsf{R})$ *consisting of two stages. Throughout, both parties receive the security parameter* 1^n *as input.*

> COMMIT. *The sender* S *has a private input* $b \in \{0,1\}$*, which it wishes to commit to the receiver* R*, and a sequence of coin tosses* σ*. At the end of this stage, both parties receive as common output a* commitment z*.*
>
> REVEAL. *Both parties receive as input a commitment* z*.* S *also receives the private input* b *and coin tosses* σ *used in the commit stage. After the interaction of* $(\mathsf{S}(b,r), \mathsf{R})(z)$*,* R *either outputs a bit, or the reject symbol* $\perp$*.*

The commitment is receiver public-coin *if the messages the receiver sends are merely the coins it flips at each round.*

For the sake of this tutorial, we focus on commitment schemes with a generic *reveal scheme: the commitment* z *is simply the transcript of the commit stage, and in the noninteractive reveal stage,* S *sends* (b, σ) *to* R*, and* R *outputs* b *if* S*, on input* b *and randomness* σ*, would have acted as the sender did in* z*; otherwise, it outputs* $\perp$*.*

Commitment schemes have two security properties. The *hiding* property informally states that, at the end of the commit stage, an adversarial receiver has learned nothing about the message m, except with negligible probability. The *binding* property states that, after the commit stage, an adversarial sender cannot produce valid openings for two distinct messages (i.e., to both 0 and 1), except with negligible probability. Both of these security properties come in two flavors—*statistical*, where we require security even against a computationally unbounded adversary, and *computational*, where we only require security against feasible (e.g., polynomial-time) adversaries.

Statistical security is preferable to computational security, but it is impossible to have commitment schemes that are both statistically hiding and statistically binding. In this tutorial, we focus on constructing statistically hiding (and computationally binding) schemes, which are closely connected to the notion of inaccessible entropy generators.

Definition 6.2 (commitment schemes)**.** *A commitment scheme* $\mathsf{Com} = (\mathsf{S}, \mathsf{R})$ *is* statistically hiding *if the following holds:*

> COMPLETENESS. *If both parties are honest, then for any bit* $b \in \{0,1\}$ *that* S *gets as private input,* R *accepts and outputs* b *at the end of the reveal stage.*
>
> STATISTICAL HIDING. *For every unbounded strategy* $\widetilde{\mathsf{R}}$*, the distributions* $\mathrm{view}_{\widetilde{\mathsf{R}}}((\mathsf{S}(0), \widetilde{\mathsf{R}})(1^n))$ *and* $\mathrm{view}_{\widetilde{\mathsf{R}}}((\mathsf{S}(1), \widetilde{\mathsf{R}})(1^n))$ *are statistically indistinguishable, where* $\mathrm{view}_{\widetilde{\mathsf{R}}}(e)$ *denotes the collection of all messages exchanged and the coin tosses of* $\widetilde{\mathsf{R}}$ *in* e*.*
>
> *The commitment is* honestreceiver *statistically hiding, if the above is only guaranteed for* $\widetilde{\mathsf{R}} = \mathsf{R}$*.*
>
> COMPUTATIONAL BINDING. *A* PPT $\widetilde{\mathsf{S}}$ *succeeds in the following game (breaks the commitment) only with negligible probability in* n*:*

[12] We present the definition for bit commitment. To commit to multiple bits, we may simply run a bit commitment scheme in parallel.

- $\widetilde{\mathsf{S}} = \widetilde{\mathsf{S}}(1^n)$ *interacts with an honest* $\mathsf{R} = \mathsf{R}(1^n)$ *in the commit stage, on security parameter* 1^n*, which yields a commitment* z*.*
- $\widetilde{\mathsf{S}}$ *outputs two messages* τ_0, τ_1 *such that* $\mathsf{R}(z, \tau_b)$ *outputs* b*, for both* $b \in \{0,1\}$*.*

Com *is* δ-$\mathsf{binding}$ *if no* PPT $\widetilde{\mathsf{S}}$ *wins the above game with probability larger than* $\delta(n) + \mathrm{neg}(n)$*.*

We now discuss the intriguing connection between statistically hiding commitment and inaccessible entropy generators. Consider a statistically hiding commitment scheme in which the sender commits to a message of length k, and suppose we run the protocol with the message m chosen uniformly at random in $\{0,1\}^k$. Then, by the statistically hiding property, the *real entropy* of the message m after the commit stage is $k - \mathrm{neg}(n)$. On the other hand, the computational binding property states that the *accessible entropy* of m after the commit stage is at most $\mathrm{neg}(n)$. This is only an intuitive connection, since we have not discussed real and accessible entropy for protocols, but only for generators. Such definitions can be found in [14], and for them it can be proven that statistically hiding commitments imply protocols in which the real entropy is much larger than the accessible entropy. Here our goal is to establish the converse, namely that a generator with a gap between its real and accessible entropy implies a statistically hiding commitment scheme. The extension of this fact for protocols can be found in [14].

Theorem 6.3 (Inaccessible entropy generator to statistically hiding commitment)**.** *Let* $k = k(n)$*,* $c = c(n)$*,* $s = s(n)$ *be polynomial-time computable functions and let* $p = p(n) \in \mathrm{poly}$*. Let* G *be an efficient* $m = m(n)$*-block generator over* $\{0,1\}^c \times \{0,1\}^s$*, and assume that* G*'s real Shannon entropy is at least* k *and its accessible entropy is at most* $(1 - 1/p) \cdot k$*. Then for any polynomial-time computable* $g = g(n) \geq \max\{\mathrm{H}_0(G(U_c, U_s)), \log m\}$*, there exists an* $O(mpg/k)$*-round, statistically hiding and computationally binding commitment scheme. Furthermore, if the bound on the real entropy is invariant to the public parameter, then the commitment is receiver public-coin.*

Given per n polynomial-size advice, the commitment round complexity can be reduced to $O(m)$; see Remark 6.7 for details. In Section 6.2 we use this fact to prove that an inaccessible generator with a *constant* number of blocks yields a constant round commitment.

Combining the above theorem with Theorem 4.2 reproves the following fundamental result:

Theorem 6.4 (One-way functions to statistically hiding commitment)**.** *Assume there exists one-way function* $f\colon \{0,1\}^n \mapsto \{0,1\}^n$*. Then there exists an* $O(n^2/\log^2 n)$*-round, receiver public-coin statistically hiding commitment scheme.*

Given per n polynomial-size advice, the round complexity of the above commitment can be reduced to $O(n/\log n)$, matching the lower bound for such fully black box constructions of [18].

The heart of the proof of Theorem 6.3 lies in the following lemma.

Lemma 6.5. *Let* $k(n) \geq 3n$ *be a polynomial-time computable function, let* G *be an efficient* m*-block generator, and assume one-way functions exist. Then for every efficiently computable* $p(n) \geq 1/\mathrm{poly}(n)$ *there exists a polynomial-time,* $O(m)$*-round, receiver public-coin, commitment scheme* Com *with the following properties:*

Hiding: *If each block of* $\mathsf{G}(U_{c(n)}, U_{s(n)})$ *has real min-entropy at least* $k(n)$*, then* Com *is statistically hiding. Furthermore, if the bound on the real entropy is invariant to the public parameter, then the commitment is receiver public-coin.*

Binding: *If for every efficient* G*-consistent, online generator* $\widetilde{\mathsf{G}}$ *and all large enough* n*,*

$$\mathsf{Pr}_{\boldsymbol{t} \xleftarrow{R} T_{\widetilde{\mathsf{G}}}(1^n)} \left[\mathrm{AccH}^{\widetilde{\mathsf{G}}}(\boldsymbol{t}) > m(n)(k(n) - 3n) \right] \leq 1 - 1/p(n),$$

then Com *is computationally binding.*

We prove Lemma 6.5 in Section 6.1, but first use it to prove Theorem 6.3.

Proving Theorem 6.3.

Proof of Theorem 6.3. We prove Theorem 6.3 by manipulating the real and accessible entropy of G using the tools described in Section 5, and then applying Lemma 6.5 on the resulting generator.

Truncated sequential repetition: real entropy equalization. In this step we use G to define a generator $\mathsf{G}^{[w]}$ that each of whose blocks has the same amount of real entropy: the average of the real entropy of the blocks of G. In relative terms, the entropy gap of $\mathsf{G}^{[w]}$ is essentially that of G.

We assume without loss of generality that $m(n)$ is a power of two.[13] Consider the efficient $m' = m'(n) = (w-1) \cdot m$)-block generator $\mathsf{G}^{[w]}$ resulting by applying truncated sequential repetition (see Definition 5.1) on G with parameter $w = w(n) = \max\{4, \lceil 12gp/k \rceil\} \leq \mathrm{poly}(n)$. By Lemma 5.2:

- *Each block* of $\mathsf{G}^{[w]}$ has real entropy at least $k' = k'(n) = k/m$.
- The accessible entropy of $\mathsf{G}^{[w]}$ is at most

$$\begin{aligned} a' = a'(n) &= (w-2) \cdot (1 - 1/p) \cdot k + \log m + 2\,\mathrm{H}_0(G(U_c, U_s))) \\ &\leq (w-2) \cdot (1 - 1/p) \cdot k + 3g \\ &= (w-2) \cdot (1 - 1/2p) \cdot k - (w-2) \cdot k/2p + 3g \\ &\leq (w-2) \cdot (1 - 1/2p) \cdot k - w \cdot k/4p + 3g \\ &\leq (w-2) \cdot (1 - 1/2p) \cdot k \\ &< m' \cdot (1 - 1/2p) \cdot k'. \end{aligned}$$

Direct product: converting real entropy to min-entropy and gap amplification. In this step we use $\mathsf{G}^{[w]}$ to define a generator $(\mathsf{G}^{[w]})^{\langle v \rangle}$ that each of whose blocks has the same amount of real min-entropy, about v times the per-block real entropy of $\mathsf{G}^{[w]}$. The accessible entropy of $(\mathsf{G}^{[w]})^{\langle v \rangle}$ is at most v times the accessible entropy of $\mathsf{G}^{[w]}$.

We assume without loss of generality that the output blocks are of all of the same length $\ell = \ell(n) \in \Omega(\log n)$.[14] Let $v = v(n) = \max\left\{32np/k', \left\lceil c \cdot (\log(n)\ell p)/k')^2 \right\rceil\right\}$ for $c > 0$ to be determined by the analysis. Consider the efficient m'-block generator $(\mathsf{G}^{[w]})^{\langle v \rangle}$, generated by taking the direct product of $\mathsf{G}^{[w]}$ according to Definition 5.4. By Lemma 5.5:

- Each block of $(\mathsf{G}^{[w]})^{\langle v \rangle}$ has real *min-entropy* at least $k'' = k''(n) = v \cdot k' - O\left(\log(n) \cdot \ell \cdot \sqrt{v}\right)$.

[13] Adding $2^{\lceil \log m(n) \rceil} - m(n)$ final blocks of constant value transforms a block-generator to one whose block complexity is a power of two, while maintaining the same amount of real and accessible entropy.

[14] A standard padding technique can be used to transform a block-generator to one whose blocks are all of the same length, without changing its real and its accessible entropy.

- The accessible entropy of $(\mathsf{G}^{[w]})^{\langle v\rangle}$ is at most $a'' = a''(n) = v \cdot a'$.

Hence for large enough n, it holds that

$$\begin{aligned} m' \cdot k'' - a'' &\geq m' \cdot \left(v \cdot k' - O\left(\log(n) \cdot \ell \cdot \sqrt{v}\right)\right) - v \cdot a' \\ &> m' \cdot \left(v \cdot k' - O\left(\log(n) \cdot \ell \cdot \sqrt{v}\right)\right) - v \cdot m' \cdot (1 - 1/2p) \cdot k' \\ &= m' \cdot v \cdot \left(k'/2p - O(\log(n) \cdot \ell/\sqrt{v})\right) \\ &\geq m' \cdot v \cdot k'/4p \\ &\geq 4m'n. \end{aligned}$$

The penultimate inequality holds by taking a large enough value of c in the definition of v. Hence, by an averaging argument for any efficient $(\mathsf{G}^{[w]})^{\langle v\rangle}$-consistent, online generator $\widetilde{\mathsf{G}}$ and all large enough n, it holds that

$$\mathsf{Pr}_{\mathbf{t} \xleftarrow{R} T_{\widetilde{\mathsf{G}}}(1^n)}\left[\mathrm{AccH}_{\widetilde{\mathsf{G}}}(\mathbf{t}) > m'(n)(k''(n) - 3n)\right] \leq 1 - 1/p'(n) \tag{24}$$

for $p'(n) = m'(n)(k''(n) - 3n)/n$.

By Theorem 7.1, the existence of G implies that of one-way functions. Hence, we can apply Lemma 6.5 with $(\mathsf{G}^{[w]})^{\langle v\rangle}$, $k = k''$ and $p = p'$, to get the claimed $(m' = m \cdot (w-1) = O(m \cdot gp/k))$-round, statistically hiding and computationally binding commitment.

Finally, it readily follows from the above proof that if the bound on the real entropy of G is invariant to the public parameter, then so is that of $(\mathsf{G}^{[w]})^{\langle v\rangle}$. Hence, Lemma 6.5 yields that in this case the resulting commitment is receiver public-coin. □

Remark 6.6 (Comparison with the construction of next-block pseudoentropy generators to pseudorandom generators)**.** *It is interesting to see the similarity between the manipulations we apply above on the inaccessible entropy generator* G *to construct statistically hiding commitment, and those applied by Haitner et al. [17], Vadhan and Zheng [30] on the next-block pseudoentropy generator to construct a pseudorandom generator. The manipulations applied in both constructions are essentially the same and achieve similar goals: to convert real entropy to per-block min-entropy whose overall sum is significantly larger than the accessible entropy in the above, and to convert next-block pseudoentropy to per-block pseudo-min-entropy whose overall sum is significantly larger than the real entropy in [17, 30]. This fact, together with the similarity in the initial steps of constructing the above generators from one-way functions (inaccessible entropy generator above and next-block pseudoentropy generator in [30]) yields that the constructions of statistically hiding commitment schemes and pseudorandom generators from one-way functions are surprisingly similar.*

Remark 6.7 (Omitting the entropy equalizing step)**.** *If the amount of real entropy of each block of* G *is* efficiently computable*, the entropy equalizing step in the proof of Theorem 6.3 above is not needed. Rather, we can take a direct product of* G *itself to get the desired generator. This argument yields an* $\Theta(m)$*-round, nonuniform (the parties use a nonuniform polynomial-size advice per security parameter) commitment scheme, assuming the bound on the accessible entropy of* G *holds for nonuniform generators. When combined with Theorem 4.2, the latter yields a* $\Theta(n/\log n)$*-round nonuniform commitment statistically hiding scheme from any nonuniform one-way function, matching the lower bound of [18].*[15]

[15]The bound of [18] is stated for uniform commitment schemes, but the same bound for nonuniform commitment schemes readily follows from their proof.

If the generator's number of blocks is constant*, the knowledge of the per-block entropy is not needed. Rather, the above reduction can be applied to all possible values for the real entropy of the blocks (up to some* $1/\text{poly}$ *additive accuracy level), yielding polynomially many commitments that are all binding and at least one of them is hiding. Such commitments can then be combined in a standard way to get a single scheme that is statistically hiding and computationally binding. See Section 6.2 for details.*

6.1 Proving Lemma 6.5

To prove Lemma 6.5, we use a random block of G to mask the committed bit. The guarantee about the real entropy of G yields that the resulting commitment is hiding, whereas the guarantee about G's accessible entropy yields that the commitment is weakly (i.e., $\Omega(1/mp)$) binding. This commitment is then amplified via parallel repetition, into a full-fledged computationally binding and statistically hiding commitment. In more detail, the construction of the aforementioned weakly binding commitment scheme goes as follows: R samples the public parameter z and sends it to S, and S starts (privately) computing a random execution of $\mathsf{G}(z,\cdot)$. At the i'th round, R tells S whether to send it the i'th block of G to R, or to use the i'th block y_i as the sender input for a (constant round) "hashing" subprotocol. This subprotocol has the following properties:

- Conditioned on $y_1,\ldots,y_{i-1}$ and the hash value of y_i (i.e., the transcript of the hashing protocol), the (real) min-entropy of y_i is still high (e.g., $\Omega(n)$), and
- if the accessible entropy of G in the i'th block is lower than $k-2n$ (i.e., given an adversarial generator view, the support size of y_i is smaller than 2^{k-2n}), then y_i is *determined* from the point of view of (even a cheating) S after sending the hash value.

Next, S "commits" to its secret bit b by masking it (via XORing) with a bit extracted (via an inner product with a random string) from y_i, and the commit stage halts.

The hiding of the above scheme follows from the guarantee about the min-entropy of G's blocks. The $1/mp$-binding of the scheme follows since the bound on the accessible entropy of G yields that with some probability, the accessible entropy of at least one of G's blocks is low, and thus the sender is bounded to a single bit if the receiver has chosen this block to use for the commitment.

The aforementioned hashing protocol is defined and analyzed in Section 6.1.1, the weakly binding commitment is defined in Section 6.1.2, and we put it all together to prove the lemma in Section 6.1.3.[16]

6.1.1 Strongly Binding Hashing Protocol

A building block of our hashing protocol is the "weakly binding" hashing protocol of Ding et al. [7].

Let $\mathcal{H}^1$ and $\mathcal{H}^2$ be function families over $\{0,1\}^\ell$.

Protocol 6.8 (Weakly binding hashing protocol $(\mathsf{S}_{\mathsf{WH}},\mathsf{R}_{\mathsf{WH}})^{\mathcal{H}_1,\mathcal{H}_2}$)**.**

S_{WH}*'s private input:* $x\in\{0,1\}^\ell$

[16] A simplified version of the somewhat complicated hashing protocol and the resulting weak commitment defined below is given in [11].

1. $\mathsf{R_{WH}}$ *sends* $h^1 \xleftarrow{R} \mathcal{H}^1$ *to* $\mathsf{S_{WH}}$.
2. $\mathsf{S_{WH}}$ *sends* $y^1 = h^1(x)$ *back to* $\mathsf{R_{WH}}$.
3. $\mathsf{R_{WH}}$ *sends* $h^2 \xleftarrow{R} \mathcal{H}^2$ *to* $\mathsf{S_{WH}}$.
4. $\mathsf{S_{WH}}$ *sends* $y^2 = h^2(x)$ *back to* $\mathsf{R_{WH}}$.

We will use two properties of the above protocol. The first, which we will use for hiding, is that if $\mathsf{S_{WH}}$ sends only k' bits to $\mathsf{R_{WH}}$ and $\mathsf{S_{WH}}$'s input x comes from a distribution of min-entropy significantly larger than k', the input of $\mathsf{S_{WH}}$ has high min-entropy conditioned on $\mathsf{R_{WH}}$'s view of the protocol (with high probability). On the other hand, the following binding property, which we refer to as "weak" to distinguish it from the binding property of the final protocol, states that if x has max-entropy smaller than k (i.e., is restricted to come from a set of size at most 2^k) and $\mathcal{H}_1$ and $\mathcal{H}_2$ are "sufficiently" independent and their total output length is sufficiently larger than k, then after the interaction ends, x will be uniquely determined, except with exponentially small probability.

The following proposition readily follows from the proof of [7, Theorem 5.6]:

Proposition 6.9 ([7], $(\mathsf{S_{WH}}, \mathsf{R_{WH}})$ is weakly binding)**.** *Let* $\mathcal{H}^1 \colon \{0,1\}^\ell \mapsto \{0,1\}^k$ *and* $\mathcal{H}^2 \colon \{0,1\}^\ell \mapsto \{0,1\}^n$ *be* ℓ*-wise and two-wise independent hash function families, respectively, and let* $\mathcal{L} \subseteq \{0,1\}^\ell$ *be a set of size at most* 2^k*. Let* $\mathsf{S}^*_{\mathsf{WH}}$ *be an (unbounded) adversary playing the role of* $\mathsf{S_{WH}}$ *in* $(\mathsf{S_{WH}}, \mathsf{R_{WH}})$ *that, following the protocol's interaction, outputs two strings,* x_0 *and* x_1*. Then, the following holds with respect to a random execution of* $(\mathsf{S_{WH}}, \mathsf{R_{WH}})^{\mathcal{H}^1, \mathcal{H}^2}$*:*

$$\mathsf{Pr}[x_0 \neq x_1 \in \mathcal{L} \ \wedge \ \forall j \in \{0,1\} : \ h^1(x_j) = y^1 \wedge h^2(x_j) = y^2] < 2^{-\Omega(n)}.$$

Namely, with save but exponentially small probability, there are no two distinct items in $\mathcal{L}$ that are consistent with the protocol transcript (i.e., with the two hash values).

Our strongly binding hashing protocol is obtained by adding *universal one-way hash functions* on top of the above protocol.[17]

Definition 6.10 (universal one-way hash functions [27])**.** *An efficient function family* $\mathcal{F} = \left\{\mathcal{F}_n = \left\{f \colon \{0,1\}^{\ell(n)} \mapsto \{0,1\}^{m(n)}\right\}\right\}_{n \in \mathbb{N}}$ *is* universal one-way (hash) *if the following holds.*

Compression. $\ell(n) > m(n)$.

Target Collision Resistance. *The probability that a* PPT A *succeeds in the following game is negligible in* n*:*

1. $(x, \mathsf{state}) \xleftarrow{R} \mathsf{A}(1^n)$
2. $f \xleftarrow{R} \mathcal{F}_n$
3. $x' \xleftarrow{R} \mathsf{A}(x, \mathsf{state}, f)$ *and* A *succeeds whenever* $x' \neq x$ *and* $f(x') = f(x)$.

[17] The following protocol is of similar flavor to (and indeed inspired by) the protocol used by Haitner and Reingold [10] in their transformation of "two-phase" commitment to statistically hiding commitment. In fact, their protocol can be seen as a special case of Protocol 6.12, designed to work for singleton sets $\mathcal{L}_v$' (see Lemma 6.13).

By Rompel [28] (full proof in [25]; see also [15]), and the length reduction of [27, Lemma 2.1], the existence of one-way functions implies that of a family of universal one-way hash functions for any poly-time computed and bounded length function ℓ.[18]

Theorem 6.11 ([28, 27, 25])**.** *Assume that one-way functions exist. Then, for any poly-time computed and bounded $\ell(n) > n$, there exists a family of universal one-way hash functions mapping strings of length $\ell(n)$ to strings of length n.*

Let $\mathcal{H}^1 = \{\mathcal{H}^1_n\}$, $\mathcal{H}^2 = \{\mathcal{H}^2_n\}$ and $\mathcal{F} = \{\mathcal{F}_n\}$ be function families over $\{0,1\}^{\ell(n)}$.

Protocol 6.12 (Strongly binding hashing protocol $(\mathsf{S_{SH}}, \mathsf{R_{SH}})^{\mathcal{H}_1,\mathcal{H}_2,\mathcal{F}}$)**.**

Common input: 1^n.

$\mathsf{S_{SH}}$*'s private input:* $x \in \{0,1\}^{\ell(n)}$.

1. *The two parties interact in* $(\mathsf{S_{WH}}(x), \mathsf{R_{WH}})^{\mathcal{H}^1_n,\mathcal{H}^2_n}$*, with* $\mathsf{S_{SH}}$ *and* $\mathsf{R_{SH}}$ *taking the role of* $\mathsf{S_{WH}}$ *and* $\mathsf{R_{WH}}$ *respectively.*
2. $\mathsf{R_{SH}}$ *sends* $f \stackrel{R}{\leftarrow} \mathcal{F}_n$ *to* $\mathsf{S_{SH}}$.
3. $\mathsf{S_{SH}}$ *sends* $y = f(x)$ *back to* $\mathsf{R_{SH}}$.

It is clear that if $\mathcal{H}^1$, $\mathcal{H}^2$ and $\mathcal{F}$ are efficient families, then the above protocol is efficient. We prove following "strong" binding property.

Lemma 6.13 ($(\mathsf{S_{SH}}, \mathsf{R_{SH}})$ is strongly binding)**.** *Let $\mathcal{H}^1 = \{\mathcal{H}^1_n\}$, $\mathcal{H}^2 = \{\mathcal{H}^2_n\}$ and $\mathcal{F} = \{\mathcal{F}_n\}$ be efficient function families, mapping strings of length $\ell(n)$ to strings of length $k(n)$, n and n, respectively. Assume $\mathcal{H}^1_n$ is $\ell(n)$-wise independent, $\mathcal{H}^2_n$ is two-wise independent, and $\mathcal{F}$ is a universal one-way hash family. Let $(\mathsf{S_{SH}}(x), \mathsf{R_{SH}})(1^n) = (\mathsf{S_{SH}}(x), \mathsf{R_{SH}})^{\mathcal{H}^1_n,\mathcal{H}^2_n,\mathcal{F}_n}(1^n)$.*

Let $\left\{\left\{\mathcal{L}^v_n \subseteq \{0,1\}^{\ell(n)}\right\}_{v\in\{0,1\}^*}\right\}_{n\in\mathbb{N}}$ *be a set ensemble with each set $\mathcal{L}^v_n$ of size at most $2^{k(n)}$. Let $\mathsf{S^*_{SH}}$ be a* PPT *adversary playing the role of $\mathsf{S_{SH}}$ in $(\mathsf{S_{SH}}, \mathsf{R_{SH}})$, outputting a string v before the interaction starts, and outputting a pair of strings (x_0, x_1) after the interaction ends. Then the following holds with respect to a random execution of $(\mathsf{S^*_{SH}}, \mathsf{R_{SH}})(1^n)$.*

$$\mathsf{Pr}\left[\begin{array}{c} x_0 \neq x_1 \ \wedge\ \{x_0, x_1\} \cap \mathcal{L}^v_n \neq \emptyset \ \wedge \\ \forall j \in \{0,1\}:\ h^1(x_j) = y^1 \ \wedge\ h^2(x_j) = y^2 \ \wedge\ f(x_j) = y \end{array}\right] = \mathrm{neg}(n),$$

where h^1, h^2, y^1 and y^2 are the corresponding values in the execution of $(\mathsf{S_{WH}}, \mathsf{R_{WH}})$ embedded in $(\mathsf{S^*_{SH}}, \mathsf{R_{SH}})$.

Namely, a cheating efficient sender committing to a small set $\mathcal{L}^v_n$ before the interaction starts cannot find two distinct strings that are consistent with the interaction and (even) one of which is in $\mathcal{L}^v_n$. This is a strengthening of the binding of Protocol 6.8 (see Proposition 6.9), which only

[18] The Target Collision Resistance property of Definition 6.10 is somewhat stronger than the one given in [25] (and somewhat weaker than the original definition in [27]). The strengthening is in allowing A to transfer additional information, i.e., state, between the selection of x and finding the collision. We note that the proof in [25] holds also with respect to our stronger definition (and even with respect to the original definition of [27]).

guarantees that there are no *two* items in a predetermined small set that are consistent with the transcript. Note, however, that the weak binding of Protocol 6.8 holds unconditionally (against any cheating strategy), where the binding of the above protocol is only guaranteed to hold against an efficient adversary. For our application of constructing statistically hiding and computationally commitment, having binding against an efficient adversary suffices.

Proof. Assume towards a contradiction that there exists a PPT $\mathsf{S}^*_{\mathsf{SH}}$ that for infinitely many n's violates the "binding" of $(\mathsf{S}_{\mathsf{SH}}, \mathsf{R}_{\mathsf{SH}})$ with success probability at least $1/p(n)$, for some $p \in \text{poly}$. Consider the following efficient algorithm for violating the target collision resistance of $\mathcal{F}_n$.

Algorithm 6.14 (Collision finder $\mathsf{ColFinder}$)**.**

Committing stage.

Input: *security parameter* 1^n.

1. *Emulate a random execution of* $(\mathsf{S}^*_{\mathsf{SH}}, \mathsf{R}_{\mathsf{SH}})(1^n)$ *until the end of the embedded execution of* $(\mathsf{S}_{\mathsf{WH}}, \mathsf{R}_{\mathsf{WH}})$*, and denote the state of the emulated protocol by* state.
2. *Continue the execution* $(\mathsf{S}^*_{\mathsf{SH}}, \mathsf{R}_{\mathsf{SH}})$ *until it ends, and let* $\{x_0, x_1\}$ *be the two values* $\mathsf{S}^*_{\mathsf{SH}}$ *outputs at the end of the emulation.*
3. *Output* (x, state)*, for* $x \xleftarrow{R} \{x_0, x_1\}$.

Finding collision.

Input: $f \in \mathcal{F}_n$, $x \in \{0,1\}^\ell$ *and* $\mathsf{state} \in \{0,1\}^*$.

1. *Emulate a random execution of* $(\mathsf{S}^*_{\mathsf{SH}}, \mathsf{R}_{\mathsf{SH}})(1^n)$ *conditioned on* state *and* f*, and let* x_0 *and* x_1 *be the output of* $\mathsf{S}^*_{\mathsf{SH}}$ *in the end of the emulation.*
2. *Output* $x' \in \{x_0, x_1\} \setminus \{x\}$.

Fix n such that $\mathsf{S}^*_{\mathsf{SH}}$ breaks the binding in $(\mathsf{S}^*_{\mathsf{SH}}, \mathsf{R}_{\mathsf{SH}})(1^n)$ with probability at least $1/p(n)$. For a given execution of $(\mathsf{S}^*_{\mathsf{SH}}, \mathsf{R}_{\mathsf{SH}})(1^n)$, let v be the string output by $\mathsf{S}^*_{\mathsf{SH}}$ before the interaction starts, and let z be the value of the element of $\mathcal{L}^v_n$ that is consistent with the embedded execution of $(\mathsf{S}_{\mathsf{WH}}, \mathsf{R}_{\mathsf{WH}})$, setting it to $\perp$ if the number of consistent elements is not one. Proposition 6.9 yields that the probability $\mathsf{S}^*_{\mathsf{SH}}$ breaks the binding and $z \neq \perp$, is at least $1/p(n) - 2^{\Omega(n)} > 1/2p(n)$.

Let State be the value of state in a random execution of $\mathsf{ColFinder}(1^n)$. For $s \in \text{Supp}(\mathsf{State})$, let $q(s)$ be the probability that $\mathsf{S}^*_{\mathsf{SH}}$ breaks the binding in $(\mathsf{S}^*_{\mathsf{SH}}, \mathsf{R}_{\mathsf{SH}})(1^n)$ and $z \neq \perp$, conditioned that its state after the execution of $(\mathsf{S}_{\mathsf{WH}}, \mathsf{R}_{\mathsf{WH}})$ is s. It is easy to verify that conditioned on $\mathsf{State} = s$, it holds that $\mathsf{ColFinder}(1^n)$ finds a collision in $\mathcal{F}_n$ (i.e., $x \neq x'$ and $f(x) = f(x')$) with probability at least $q(s)^2/2$. Hence, $\mathsf{ColFinder}(1^n)$ finds a collision with probability at least $\mathsf{E}\left[q(\mathsf{State})^2/2\right]$. By the Jensen inequality, the latter is at least $\mathsf{E}\left[q(\mathsf{State})\right]^2/2 \geq 1/8p(n)^2$, in contradiction to the target collision resistance of $\mathcal{F}_n$. □

6.1.2 Constructing Weakly Binding Commitment

We are finally ready to define the weakly binding commitment. Let $\mathsf{G}\colon \{0,1\}^{c(n)} \times \{0,1\}^{s(n)} \mapsto (\{0,1\}^{\ell(n)})^{m(n)}$ be an m-block generator. Let $\mathcal{H}^1 = \{\mathcal{H}^1_n\}$, $\mathcal{H}^2 = \{\mathcal{H}^2_n\}$ and $\mathcal{F} = \{\mathcal{F}_n\}$ be function families, mapping strings of length $\ell(n)$ to strings of length $k(n) - 3n$, n, and n, respectively. The weakly binding commitment is defined as follows:

Protocol 6.15 (Weakly binding, receiver public-coin commitment scheme $\mathsf{Com} = (\mathsf{S}, \mathsf{R})$)**.**

Common input: *security parameter* 1^n

S's private input: $b \in \{0,1\}$

Commit stage:

1. R *samples* $z \xleftarrow{R} \{0,1\}^{c(n)}$ *and send it to* S*.*
2. S *starts (internally) an execution of* $\mathsf{G}(z,x)$ *for* $x \xleftarrow{R} \{0,1\}^{s(n)}$*.*
3. *For* $i = 1$ *to* m*, the parties do the following:*
   (a) *The two parties interact in* $(\mathsf{S}_{\mathsf{SH}}(y_i = \mathsf{G}(z,x)_i), \mathsf{R}_{\mathsf{SH}})^{\mathcal{H}^1,\mathcal{H}^2,\mathcal{F}}(1^n)$*, with* S *and* R *taking the roles of* S_{SH} *and* R_{SH}*, respectively.*
   (b) R *flips a coin* c_i *to be one with probability* $1/(m+1-i)$*, and sends it to* S*.*

   If $c_i = 0$*,* S *sends* y_i *to* R*.*

   Otherwise,
   i. S *samples* $u \xleftarrow{R} \{0,1\}^{\ell(n)}$ *and sends* $(\langle u, y_i\rangle_2 \oplus b, u)$ *to* R*, for* $\langle\cdot,\cdot\rangle_2$ *being inner product modulo* 2*.*
   ii. The parties end the execution.

Assuming G is efficient and that $\mathcal{H}_1$, $\mathcal{H}_2$ and $\mathcal{F}$ are efficiently computable (i.e., sampling and evaluation time are polynomial in n), then clearly Com is an efficient (poly-time computable) correct public-message commitment scheme. It is left to prove the hiding and binding properties of Com.

In the following let i^* be the round for which c_i takes the value 1. Note that i^* is uniform over $[m]$.

Claim 6.16 (Statistically hiding)**.** *Assume each block of* $\mathsf{G}(1^n)$ *has real min-entropy at least* $k(n)$*. Then* Com *is honest-receiver statistically hiding. If the bound on the real min-entropy is invariant to the public parameter, then* Com *is statistically hiding.*

Proof. Fix $n \in \mathbb{N}$ and omit it from the notation when clear from the context. For $i \in [m]$, let Y_i denote the i'th block of $\mathsf{G}(Z,X)$ for $(Z,X) \leftarrow \{0,1\}^c \times \{0,1\}^s$. By assumption, $\mathsf{Pr}_{\boldsymbol{y} \xleftarrow{R} Y_{1,\ldots,i}}\left[\mathrm{H}_{Y_{i^*}|Z,Y_{<i}}(\boldsymbol{y}_{i^*}|z,\boldsymbol{y}_{<i}) < k\right] = \mathrm{neg}(n)$. Thus Lemma 2.4 yields that there exists a random variable Y'_i such that

1. $(Z, Y_{<i}, Y'_i)$ is statistically indistinguishable from $(Z, Y_{<i}, Y_i)$, and
2. $\mathrm{H}_\infty(Y_i|_{Z=z,Y_{<i}=\boldsymbol{y}}) \geq k$ for every $(z,\boldsymbol{y}) \in \mathrm{Supp}(Z, Y_{<i})$.

Let $\widetilde{\mathsf{R}}$ be an arbitrary algorithm playing the role of R in Com that samples z as instructed (i.e., uniformly in $\{0,1\}^c$). Let $\widetilde{Y}_1, \ldots, \widetilde{Y}_{i^*}$ be the first i^* computed by S in a random execution of $(\mathsf{S}, \widetilde{\mathsf{R}})$, and let $V^{\widetilde{\mathsf{R}}}$ be $\widetilde{\mathsf{R}}$'s view right after S sent $\widetilde{Y}_{i^*-1}$ (all variables are arbitrarily set if the execution has aborted). Since, by assumption, $\widetilde{\mathsf{R}}$ samples z uniformly and since $V^{\widetilde{\mathsf{R}}}_{i^*}$ is a probabilistic function of the public parameter Z and $\widetilde{Y}_{<i^*}$, there exists a random variable $\widetilde{Y}'_{i^*}$ such that

1. $(V^{\widetilde{\mathsf{R}}}, \widetilde{Y}_{i^*})$ is statistically indistinguishable from $(V^{\widetilde{\mathsf{R}}}, \widetilde{Y}'_{i^*})$, and
2. $\mathrm{H}_\infty(\widetilde{Y}'_{i^*}|_{V^{\widetilde{\mathsf{R}}}=v}) \geq k$, for every non-aborting view $v \in \mathrm{Supp}(V^{\widetilde{\mathsf{R}}})$.

Let W be the messages sent by S in the embedded execution of the interactive hashing $(\mathsf{S}, \widetilde{\mathsf{R}})$. Since $|W| = k - n$, by Lemmas 2.3 and 2.4 there exists a random variable $\widetilde{Y}''_{i^*}$ such that

1. $(V^{\widetilde{\mathsf{R}}}, W, \widetilde{Y}_{i^*})$ is $(\mathrm{neg}(n) + 2^{-\Omega(n)})$-close to $(V^{\widetilde{\mathsf{R}}}, W, \widetilde{Y}''_{i^*})$, and
2. $\mathrm{H}_\infty(\widetilde{Y}''_{i^*}|_{V^{\widetilde{\mathsf{R}}}=v, W=w}) \geq n/2$, for every non-aborting view $v \in \mathrm{Supp}(V^{\widetilde{\mathsf{R}}})$ and $w \in \mathrm{Supp}(W)$.

Let $V^{\widetilde{\mathsf{R}}}_b$ denotes $\widetilde{\mathsf{R}}$'s view at the *end* of the commit stage of $(\mathsf{S}(b), \widetilde{\mathsf{R}})$. By the above observation, the leftover hash lemma (Lemma 2.13) and the two-universality of the family $\{h_u(y) = \langle u, y \rangle_2 \colon u \in \{0,1\}^n\}$, it holds that $V^{\widetilde{\mathsf{R}}}_0$ and $V^{\widetilde{\mathsf{R}}}_1$ are statistically indistinguishable.

It is clear by the above analysis that if the bound on the real min-entropy is invariant to the public parameter, then the hiding holds for any $\widetilde{\mathsf{R}}$ (that might choose the public parameter arbitrarily). □

Claim 6.17 (Weak computational binding)**.** *Assume* G *is efficient and that for every efficient* G*-consistent, online generator* $\widetilde{\mathsf{G}}$ *and all large enough* n, $\mathsf{Pr}_{\boldsymbol{t} \xleftarrow{R} T_{\widetilde{\mathsf{G}}}(1^n)}\left[\mathrm{AccH}_{\widetilde{\mathsf{G}}}(\boldsymbol{t}) > m(k-3n)\right] \leq 1 - 1/p$*. Assume further that* $\mathcal{F}$ *is a family of universal one-way hash functions, that* $\mathcal{H}_1$ *and* $\mathcal{H}_2$ *are efficiently computable, and are* ℓ*-wise and two-wise independent, respectively. Then* Com *is* $(1 - 1/3mp)$*-binding.*

The proof of Claim 6.17 immediately follows from the next two claims.

Definition 6.18 (Non-failing senders)**.** *A sender* $\widetilde{\mathsf{S}}$ *is called* non-failing *with respect to a commitment scheme* (S, R) *if the following holds. Let* Z *be the transcript of the commit stage of* $(\widetilde{\mathsf{S}}, \mathsf{R})(1^n)$*, and let* Σ *be the first decommitment string that* $\widetilde{\mathsf{S}}$ *outputs in the (generic) reveal stage. Then* $\mathsf{Pr}\left[\mathsf{R}(Z, \Sigma) = \perp\right] = 0$.

That is, a non-failing sender never fails to justify its actions in the commit stage.

Claim 6.19 (Weak computational binding against non-failing senders)**.** *Let* G*,* Com*,* $\mathcal{F}$*,* $\mathcal{H}_1$ *and* $\mathcal{H}_2$ *be as in Claim 6.17. Then* Com *is* $(1 - 1/2mp))$*-binding against* non-failing *senders.*

Claim 6.20. *Assume a receiver public-coin commitment scheme is* α*-binding against non-failing senders. Then it is* $(\alpha + \mathrm{neg})$*-binding.*

Proving Claim 6.19.

Proof. Assume towards a contradiction that there exists a non-failing PPT sender $\widetilde{\mathsf{S}}$ that breaks the $(1 - 1/2mp)$-binding of Com. We use $\widetilde{\mathsf{S}}$ to construct an efficient, G-consistent generator $\widetilde{\mathsf{G}}$ that breaks the assumed bound on the accessible entropy of G. We assume for simplicity that $\widetilde{\mathsf{S}}$ is deterministic.

Fix $n \in \mathbb{N}$ for which $\widetilde{\mathsf{S}}$ breaks the binding with probability at least $1 - 1/2mp(n)$, and omit n from the notation when clear from the context. The following generator uses the ability of $\widetilde{\mathsf{S}}$ to break the binding of the embedded hashing protocol at all rounds, induced by its high probability of breaking the binding, to output high sample-entropy transcript.

Algorithm 6.21 ($\widetilde{\mathsf{G}}$—High entropy generator from cheating sender $\widetilde{\mathsf{S}}$)**.**

Security parameter: 1^n.

Input: public parameter z.

Operation:

1. *Start a random execution of* $(\widetilde{\mathsf{S}}, \mathsf{R})(1^n)$ *with* R*'s first message set to* z *and* $i^* = m$.
2. *For* $i = 1$ *to* $m - 1$*: output the value of* y_i *sent by* $\widetilde{\mathsf{S}}$ *at round* i *(as the* i*'th output block).*
3. *Continue the emulation of* $(\widetilde{\mathsf{S}}, \mathsf{R})$ *until its end. Let* $\sigma = (\cdot, r)$ *be the first decommitment string output by* $\widetilde{\mathsf{S}}$*. Output* $\mathsf{G}(r)_m$ *as the* m*'th output block.*

The efficiency of $\widetilde{\mathsf{G}}$ is clear, and since $\widetilde{\mathsf{S}}$ is non-failing, it is also clear that $\widetilde{\mathsf{G}}$ is G-consistent. In the rest of the proof we show that the computational binding of $(\mathsf{S}_{\mathsf{SH}}, \mathsf{R}_{\mathsf{SH}})^{\mathcal{H}^1,\mathcal{H}^2,\mathcal{F}}$ yields that $\widetilde{\mathsf{G}}$ violates the assumed bounds on the accessible entropy of $\widetilde{\mathsf{G}}$.

Let $T = (Z, R_1, Y_1, \ldots, R_m, Y_m) = T_{\widetilde{\mathsf{G}}}(1^n)$. That is, R_i are the coins R uses in the i'th round of the above emulation, i.e., its coins used in the i'th invocation of $(\mathsf{S}_{\mathsf{SH}}, \mathsf{R}_{\mathsf{SH}})$.

For $\mathbf{t} = (z, r_1, y_1, \ldots) \in \mathrm{Supp}(T)$ and $i \in [m]$, let $\mathcal{L}_{\mathbf{t},i}$ be the set of all low-entropy i'th block of $\widetilde{\mathsf{G}}$ given $r_{<i}$. That is,

$$\mathcal{L}_{\mathbf{t},i} = \mathcal{L}_{z,r_{<i}} := \left\{ y \colon \mathrm{H}_{Y_i|Z,R_{<i}}(z, y | r_{<i}) \leq k - 3n \right\} \tag{25}$$

We conclude the proof by showing that

$$\mathsf{Pr}_{\mathbf{t}=(\ldots,y_i,\ldots)\leftarrow T}\left[\exists i \in [m] \colon y_i \in \mathcal{L}_{\mathbf{t},i} \right] < 1/p \tag{26}$$

in contradiction to the assumed bound on the accessible entropy of G.

Assuming Equation (26) does not hold, we show that the assumption about the success probability of $\widetilde{\mathsf{S}}$ yields an algorithm for breaking the computational binding of $(\mathsf{S}_{\mathsf{SH}}, \mathsf{R}_{\mathsf{SH}}) = (\mathsf{S}_{\mathsf{SH}}, \mathsf{R}_{\mathsf{SH}})^{\mathcal{H}^1,\mathcal{H}^2,\mathcal{F}}$. The idea is that when $y_i \in \mathcal{L}_{r_{\leq i}}$, then for breaking the commitment for $i^* = i$, the cheating sender $\widetilde{\mathsf{S}}$ has to break the binding of $(\mathsf{S}_{\mathsf{SH}}, \mathsf{R}_{\mathsf{SH}})$ with respect to the small, by definition, set $\mathcal{L}_{r_{\leq i}}$.

For $i \in [m]$ and $(z, r_{\leq i}) \in \mathrm{Supp}(Z, R_{\leq i})$, consider the execution of $(\widetilde{\mathsf{S}}, \mathsf{R})(1^n)$ induced by z, $r_{\leq i}$ and $i^* = i$: the coins used by R in the first j'th execution of $(\mathsf{S}_{\mathsf{SH}}, \mathsf{R}_{\mathsf{SH}})$, for $j \in [i]$, are set to r_j, $c_1 = \ldots = c_{i-1} = 0$ and $c_i = 1$.

Let $\widehat{Y}_{z,r_{\leq i},0} = \mathsf{G}(z,s_0)_i$, for $\tau_0 = (s_0,\cdot,0), \tau_1 = (s_1,\cdot,1)$ being the two strings output by $\widetilde{\mathsf{S}}$ at the end of the above interaction. Note that since $\widetilde{\mathsf{S}}$ is non-failing, s_0 is always consistent with the interaction, and in particular $\widehat{Y}_{z,r_{\leq i},0}$ is well defined. Similarly, if τ_1 is a valid decommitment, let $\widehat{Y}_{z,r_{\leq i},1} = \mathsf{G}(z,s_1)_i$; otherwise, let $\widehat{Y}_{z,r_{\leq i},1} = \widehat{Y}_{z,r_{\leq i},0}$.

Let $\mathsf{low}(\mathbf{t})$ be the smallest value of i for which $y_i \in \mathcal{L}_{z,r_{\leq i}}$, set to $\perp$ if there is no such i, and let $L = \mathsf{low}(T)$. The assumption that Equation (25) does not holds implies that $\mathsf{Pr}\left[L \neq \perp\right] \geq 1/p$. Thus,

$$\mathsf{Pr}\left[\widehat{Y}_{Z,R_{\leq L},0} \neq \widehat{Y}_{Z,R_{\leq L},1} \in \{0,1\}^{\ell} \mid L \neq \perp\right] \geq 1/2 \tag{27}$$

Indeed, if Equation (27) does not hold, then $\widetilde{\mathsf{S}}$ fails to break the commitment with probability at least $\mathsf{Pr}\left[L = i^*\right] \cdot 1/p = \mathsf{Pr}\left[L \neq \perp\right] \cdot 1/m \cdot 1/2 \geq 1/2mp$, in contradiction to the assumed success probability of $\widetilde{\mathsf{S}}$. It follows that

$$\mathsf{Pr}\left[\widehat{Y}_{Z,R_{\leq L},0} \neq \widehat{Y}_{Z,R_{\leq L},1} \in \{0,1\}^{\ell} \wedge Y_L \in \mathcal{L}_{R_{\leq L}}\right] \geq \mathsf{Pr}\left[L \neq \perp\right] \cdot 1/2 \geq 1/2p \tag{28}$$

We conclude the proof by using the above observation to define an algorithm for breaking the computational binding of $(\mathsf{S}_{\mathsf{SH}}, \mathsf{R}_{\mathsf{SH}})$.

Algorithm 6.22 (Algorithm $\mathsf{S}^*_{\mathsf{SH}}$ for breaking the binding of $(\mathsf{S}_{\mathsf{SH}}, \mathsf{R}_{\mathsf{SH}})$.)**.**

Input: *security parameter* 1^n.

1. *Sample* $\boldsymbol{t} = (z, r_1, \ldots) \leftarrow T_{\widetilde{\mathsf{G}}}(1^n)$ *and* $i \xleftarrow{R} [m = m(n)]$. *Output* $v = r_{<i}$.
2. *Emulate* $(\widetilde{\mathsf{S}}, \mathsf{R})(1^n)$ *for its first* $(i-1)$ *rounds, with* R*'s first message set to* z*, and the coins used by* R *in the first* j*'th execution of* $(\mathsf{S}_{\mathsf{SH}}, \mathsf{R}_{\mathsf{SH}})$*, for* $j \in [i-1]$*, are set to* r_j*, and* $c_1 = \ldots = c_{i-1} = 0$.
3. *Interact in* $(\mathsf{S}_{\mathsf{SH}}, \mathsf{R}_{\mathsf{SH}})^{\mathcal{H}^1,\mathcal{H}^2,\mathcal{F}}(1^n)$*, by forwarding* R*'s messages to* $\widetilde{\mathsf{S}}$*, and* $\widetilde{\mathsf{S}}$*'s answers back to* R.
4. *Send* $c_i = 1$ *to* $\widetilde{\mathsf{S}}$. *Let* $(\tau_0 = (s_0,\cdot,0), \tau_1 = (s_1,\cdot,1))$ *be the two decommitment strings output by* $\widetilde{\mathsf{S}}$.
5. *Set* $y_{i,0} = \mathsf{G}(z,s_0)_i$. *If* τ_1 *is a valid decommitment set* $y_{i,1} = \mathsf{G}(z,s_1)_i$*; otherwise, set* $y_{i,1} = y_{i,0}$.
6. *If* $i = m$*, let* $y_i = y_{i,0}$.
   Otherwise,
   (a) *Rewind* $\widetilde{\mathsf{S}}$ *to its state just before it received the message* $c_i = 1$ *above.*
   (b) *Send* $c_i = 0$ *to* $\widetilde{\mathsf{S}}$. *Let* y_i *be the next message sent by* $\widetilde{\mathsf{S}}$.
7. *Output* $x_0 = y_i$ *and* $x_1 \leftarrow \{y_{i,0}, y_{i,1}\}$.

Since $\widetilde{\mathsf{S}}$ is non-failing, the pair (x_0, x_1) output by $\mathsf{S}^*_{\mathsf{SH}}$ is always consistent with its interaction with R_{SH} (happens in Step 3). In addition, for infinitely many n's, it holds that

$$\begin{aligned}
\Pr\left[x_0 \in \mathcal{L}_v \wedge x_0 \neq x_1\right] &\geq \Pr\left[i = \mathsf{low}(\mathbf{t})\right] \cdot \Pr\left[x_0 \in \mathcal{L}_{z,r_{\leq i}} \wedge x_0 \neq x_1 \mid i = \mathsf{low}(\mathbf{t})\right] \\
&\geq \frac{1}{mp} \cdot \frac{1}{2} \cdot \Pr\left[x_0 \in \mathcal{L}_{z,r_{\leq i}} \mid i = \mathsf{low}(\mathbf{t})\right] \cdot \Pr\left[y_{i,0} \neq y_{i,1} \mid i = \mathsf{low}(\mathbf{t})\right] \\
&\geq \frac{1}{2mp} \cdot \frac{1}{2} \cdot 1 \cdot \Pr\left[y_{i,0} \neq y_{i,1} \mid i = \mathsf{low}(\mathbf{t})\right] \\
&\geq \frac{1}{4mp} \cdot \frac{1}{2} = \frac{1}{8mp}.
\end{aligned}$$

The last inequality is due to Equation (28). Since by definition $\left|\mathcal{L}_{r_{\leq i}}\right| \leq 2^{k(n)-3n}$, algorithm $\mathsf{S}^*_{\mathsf{SH}}$ violates the soundness of $(\mathsf{S}_{\mathsf{SH}}, \mathsf{R}_{\mathsf{SH}})^{\mathcal{H}^1,\mathcal{H}^2,\mathcal{F}}$ guaranteed by Lemma 6.13. □

Proving Claim 6.20.

Proof. Let $\mathsf{Com} = (\mathsf{S}, \mathsf{R})$ be a receiver public-coin commitment scheme, and assume there exists an efficient cheating sender $\widetilde{\mathsf{S}}$ that breaks the binding of Com with probability at least $\alpha(n) + 1/p(n)$, for some $p \in$ poly and infinitely many n's. We construct an efficient non-failing sender $\widehat{\mathsf{S}}$ that breaks the binding of Com with probability $\alpha(n) + 1/2p(n)$, for infinitely many n's. It follows that if Com is $\alpha(n)$-binding for non-failing senders, then it is $(\alpha(n) + \mathrm{neg}(n))$-binding.

We assume for simplicity that $\widetilde{\mathsf{S}}$ is deterministic, and define the non-failing sender $\widehat{\mathsf{S}}$ as follows: $\widehat{\mathsf{S}}$ starts acting as $\widetilde{\mathsf{S}}$, but before forwarding the i'th message y_i from $\widetilde{\mathsf{S}}$ to R, it first makes sure it will be able to "justify" this message — to output an input for S that is consistent with y_i, and the message $y_1, \ldots, y_{i-i}$ it sent in the previous rounds. To find such a justification string, $\widehat{\mathsf{S}}$ continues, in its head, the interaction between the emulated $\widetilde{\mathsf{S}}$ and R until its end, using fresh coins for the receiver's messages. Since the receiver is public-coin, this efficient random continuation has the same distribution as a (real) random continuation of $(\widetilde{\mathsf{S}}, \mathsf{R})$ has. The sender $\widehat{\mathsf{S}}$ applies such random continuations polynomially many times, and if following one of them $\widetilde{\mathsf{S}}$ outputs a valid decommitment string (which by definition is a valid justification string), it keeps it for future use, and outputs y_i as its i'th message. Otherwise (i.e., it failed to find a justification string for y_i), $\widehat{\mathsf{S}}$ continues as the honest S whose coins and input bit are set to the justification string $\widehat{\mathsf{S}}$ found in the previous round.

Since $\widehat{\mathsf{S}}$ maintains the invariant that it can always justify its messages, it can also do that at the very end of the commitment stage, and thus outputting this string makes it a non-failing sender. In addition, note that $\widehat{\mathsf{S}}$ only fails to find a justification string if $\widetilde{\mathsf{S}}$ has a very low probability to open the commitment at the end of the current interaction, and thus very low probability to cheat. Hence, deviating from $\widetilde{\mathsf{S}}$ on such transcripts will only slightly decrease the cheating probability of $\widehat{\mathsf{S}}$ compared with that of $\widetilde{\mathsf{S}}$.

Assume for concreteness that R sends the first message in Com. The non-failing sender $\widehat{\mathsf{S}}$ is defined as follows:

Algorithm 6.23 (Non-failing sender $\widehat{\mathsf{S}}$ from failing sender $\widetilde{\mathsf{S}}$)**.**

Input: 1^n

Operation:

1. *Set* $w = (0^{s(n)}, 0)$*, for* $s(n)$ *being a bound on the number of coins used by* S*, and set* $\mathsf{Fail} = \mathsf{false}$.
2. *Start an execution of* $\widetilde{\mathsf{S}}(1^n)$.
3. *Upon getting the* i*'th message* q_i *from* R*, do:*
   (a) *If* $\mathsf{Fail} = \mathsf{false}$,
      i. *Forward* q_i *to* $\widetilde{\mathsf{S}}$*, and continue the execution of* $\widetilde{\mathsf{S}}$ *until it sends its* i*'th message.*
      ii. // Try and get a justification string for this i'th message.
      Do the following for $3np(n)$ *times:*
         A. *Continue the execution of* $(\widetilde{\mathsf{S}}, \mathsf{R})$ *until its end, using uniform random messages for* R.
         B. *Let* z' *and* w' *be the transcript and first message output by* $\widetilde{\mathsf{S}}$*, respectively, at the end of this execution.*
         C. *Rewind* $\widetilde{\mathsf{S}}$ *to its state right after sending its* i*'th message.*
         D. // Update the justification string.
         If $\mathsf{R}(z', w') \neq \perp$*. Set* $w = w'$ *and break the loop.*
      iii. *If the maximal number of attempts has been reached, set* $\mathsf{Fail} = \mathsf{true}$.
   (b) // Send the i'th message to R.
   If $\mathsf{Fail} = \mathsf{false}$, this will be the message sent by $\widetilde{\mathsf{S}}$ in Step $3(a)$. Otherwise, the string will be computed according to the justification string found in a previous round.
   Send a_i *to* R*, for* a_i *being the* i*'th message that* $\mathsf{S}(1^n, w)$ *sends to* R *upon getting the first* i *messages sent by* R.
4. *If* $\mathsf{Fail} = \mathsf{false}$*, output the same value that* $\widetilde{\mathsf{S}}$ *does at the end of the execution.*
   Otherwise, output w.

It is clear that $\widehat{\mathsf{S}}$ is non-failing and runs in polynomial time. It is left to show that it breaks the binding of Com with high enough probability. We do that by coupling a random execution of $(\widehat{\mathsf{S}}, \mathsf{R})$ with that of $(\widetilde{\mathsf{S}}, \mathsf{R})$, by letting R send the same, uniformly chosen, messages in both executions. We will show that the probability that $\widetilde{\mathsf{S}}$ breaks the binding, but $\widehat{\mathsf{S}}$ fails to do so, is at most $1/3p(n) + m \cdot 2^{-n}$, for m being the round complexity of Com. If follows that, for infinitely many n's, $\widehat{\mathsf{S}}$ breaks the binding of Com with probability $\alpha(n) + 1/2p(n)$.

Let δ_i denote the probability of $\widetilde{\mathsf{S}}$ to break the binding after sending its i'th message, where the probability is over the messages to be sent by R in the next rounds. By definition of $\widehat{\mathsf{S}}$, the probability that $\delta_i \geq 1/3p(n)$ for all $i \in [m]$, and yet $\widehat{\mathsf{S}}$ set $\mathsf{Fail} = \mathsf{true}$, is at most $m \cdot 2^{-n}$. We conclude that the probability that $\widehat{\mathsf{S}}$ does not break the commitment, and yet $\widetilde{\mathsf{S}}$ does, is at most $1/2p(n) + m \cdot 2^{-n}$. □

6.1.3 Putting it Together

Given the above, we prove Lemma 6.5 as follows:

Proof of Lemma 6.5. We use efficient ℓ-wise function family $\mathcal{H}^1 = \left\{\mathcal{H}^1_n\right\}$ and two-wise function family $\mathcal{H}^2 = \left\{h^2_n\right\}_{n\in\mathbb{N}}$ mapping strings of length $\ell(n)$ to strings of length $k(n) - 3n$ and n, respectively (see [5, 6] for constructions of such families). Since, by assumption, one-way functions exist, we use Theorem 6.11 to construct universal hash function families $\mathcal{F}$ mapping strings of length $\ell(n)$ to strings of length n.

Claims 6.16 and 6.17 yield that the invocation of Protocol 6.15 with the generator G and the above function families is an $O(m)$-round, receiver public-coin commitment scheme Com that is honest-receiver statistically hiding if the real entropy of G is sufficiently large, and is $(1-\Theta(1/mp))$-binding if the generator accessible entropy is sufficiently small. Let $t = \log(n)^2 pm$ be an efficiently computable function and let $\mathsf{Com}^{\langle t\rangle} = (\mathsf{S}^{\langle t\rangle}, \mathsf{R}^{\langle t\rangle})$ be the t parallel repetition of Com: an execution of $(\mathsf{S}^{\langle t\rangle}(b), \mathsf{R}^{\langle t\rangle})(1^n)$ consists of t-fold parallel and independent executions of $(\mathsf{S}(b), \mathsf{R})(1^n)$. It is easy to see that since Com is honest receiver statistically hiding, so is $\mathsf{Com}^{\langle t\rangle}$. Finally, since Com is $(1 - \Theta(1/pm))$-binding, by [20] (recall that Com is receiver public coin) $\mathsf{Com}^{\langle t\rangle}$ is computationally binding. By [12, Cor 6.1], this yields an $O(m)$-round statistically hiding protocol.

Assuming the bound on the real entropy of G is invariant to the public parameter, Claim 6.16 yields that $\mathsf{Com}^{\langle t\rangle}$ is already statistically hiding. Thus, we do not have to use the (non-public-coin) reduction of [12], and immediately get a receiver public coin statistically hiding commitment.

□

6.2 Constant-Round Commitments

In this section we prove that an inaccessible entropy generator of constant number of blocks yields a constant-round statistically hiding commitment.

Theorem 6.24 (Inaccessible entropy generator to statistically hiding commitment, constant-round version)**.** *Let G be an efficient block-generator with a constant number of blocks. Assume G's real Shannon entropy is at least $k(n)$ for some efficiently computable function k, and that its accessible entropy is bounded by $k(n) - 1/p(n)$ for some $p \in \text{poly}$. Then there exists a constant-round statistically hiding and computationally binding commitment scheme. Furthermore, if the bound on the real entropy is invariant to the public parameter, then the commitment is receiver public-coin.*

The heart of the proof of Theorem 6.24 lies in the following lemma. In the following we use the natural generalization of commitment schemes for nonuniform protocols: the correctness, binding and hiding hold for adversaries seeing the nonuniform advice.

Lemma 6.25. *Let G be an efficient block-generator with a constant number of blocks m and block length $\ell = \ell(n)$, and assume one-way functions exist. Then for every efficiently computable $p(n) \geq 1/\text{poly}(n)$ there exists a polynomial-time, $O(m)$-round, commitment scheme Com such that the following holds for any polynomial size $\left\{\widetilde{k}_n = (\widetilde{k}_n(1), \ldots, \widetilde{k}_n(m))\right\}_{n\in\mathbb{N}}$.*

Correctness: *$\mathsf{Com}(1^n, \widetilde{k}_n)$ is correct.*

Hiding: *If for each $n \in \mathbb{N}$ and $i \in [m]$, either $\widetilde{k}_n(i) = 0$ or the i'th block of $\mathsf{G}(U_{c(n)}, U_{s(n)})$ has real min-entropy at least $\widetilde{k}_n(i) \geq 3n$, then $\mathsf{Com}(1^n, \widetilde{k}_n)$ is statistically hiding. Furthermore, if the bound on the real entropy is invariant to the public parameter, then the commitment is receiver public-coin.*

Binding: *If for every efficient* G*-consistent, online generator* $\widetilde{\mathsf{G}}$ *and all large enough* n*,*

$$\mathsf{Pr}_{\boldsymbol{t}\xleftarrow{R}T_{\widetilde{\mathsf{G}}}(1^n)}\left[\mathrm{AccH}_{\widetilde{\mathsf{G}}}(\boldsymbol{t}) > \left(\sum_{i\in[m]}\widetilde{k}_n(i)\right) - 3mn\right] \leq 1 - 1/p(n),$$

then $\mathsf{Com}(1^n, \widetilde{k}_n)$ *is computationally binding.*

Proof. The proof of Lemma 6.25 follows the same line as the proof of Lemma 6.5, where in the i'th round, the $\ell(n)$-wise independent hash function outputs $\widetilde{k}_n(i) - n$ bits (rather than $k(n) - n$ as in Lemma 6.5). If $\widetilde{k}_n(j) = 0$, the j'th block is skipped. The correctness and hiding are clear, and the binding holds since the nonuniform advice $\widetilde{k}_n$ is of logarithmic size and thus only improves the probability of outputting a high entropy block by a (fixed) polynomial factor (an improvement that can be accommodated by increasing the number of repetitions of the weakly binding commitment). □

Given Lemma 6.25, the proof of Theorem 6.24 is similar to that of Theorem 6.3 adapted to exploit the constant number of blocks.

Proof of Theorem 6.24. Let $c = c(n)$ and $s = s(n)$ be the pubic parameter and input length of G. We assume without loss of generality that the output blocks are of all of the same length $\ell = \ell(n) \in \Omega(\log n)$. We also assume for simplicity that k, the bound on the real entropy of G is an integer function. For $n \in \mathbb{N}$, let $Z_n \xleftarrow{R} \{0,1\}^c$ and let $(Y_1(n), \ldots, Y_m(n)) = \mathsf{G}(Z_n, U_{s(n)})$, and for $i \in [m]$, let $f_n(i) = \left\lfloor \mathrm{H}_{Y_i(n)|Z,Y_{<i}(n)} \right\rfloor_{1/2mp}$, for $\lfloor x \rfloor_\delta := \lfloor x/\delta \rfloor \cdot \delta$, and let $f = f(n) = \sum_{i\in[m]} f_n(i)$. By definition, $f(n) \geq k(n)(1 - 1/2p(n))$.

In the following we omit n when clear from the context. Let $\mathsf{G}^{\langle v \rangle}$ be the direct product of G (see Definition 5.4), with $v = v(n) = \max\left\{32nmp, \left\lceil c \cdot (\log n \cdot \ell mp)^2 \right\rceil\right\}$, for $c > 0$ to be determined by the analysis. By Lemma 5.5:

- The i'th block of $\mathsf{G}^{\langle v \rangle}$ has real *min-entropy* at least $f_n(i) = v \cdot f_n(i) - c' \log(n) \cdot \ell \cdot \sqrt{v}$ for some universal constant c'. In particular (by taking large enough c in the definition of v), $f_n(i) > 0$ implies $f'_n(i) \geq 3n$.
- The accessible entropy of $\mathsf{G}^{\langle v \rangle}$ is at most $a' = v \cdot (k - 1/p)$.

Let $f' = f'(n) = \sum_{i\in[m]} f'_n(i)$. The above yields that for large enough n,

$$\begin{aligned} f' - a' &\geq v \cdot \left(k(1 - 1/2p) - O\left(\log(n) \cdot \ell/\sqrt{v}\right) - k(1 - 1/p)\right) \\ &\geq vk/4p \\ &\geq 4mn. \end{aligned} \tag{29}$$

The penultimate inequality holds by taking a large enough value of c in the definition of v. Hence by an averaging argument, for any efficient $\mathsf{G}^{\langle v \rangle}$-consistent, online generator $\widetilde{\mathsf{G}}$ and all large enough n, it holds that

$$\mathsf{Pr}_{\mathbf{t}\xleftarrow{R}T_{\widetilde{\mathsf{G}}}(1^n)}\left[\mathrm{AccH}_{\widetilde{\mathsf{G}}}(\mathbf{t}) > f' - 3mn\right] \leq 1 - 1/p'(n) \tag{30}$$

for $p'(n) = (f'(n) - 3mn)/n$.

For $n \in \mathbb{N}$, let $\mathcal{K}_n$ be the set of all possible values for k_n, i.e., all tuples of the form $\widetilde{k}_n = (\widetilde{k}[1], \ldots, \widetilde{k}[m])$ with $\widetilde{k}[i] \in \left\{0, 1/2mp, \ldots, \lfloor k(n) \rfloor_{1/2mp}\right\}$ and $\sum_{i\in[m]} \widetilde{k}[i] \in [k(n)(1-1/2p(n), k(n)]$. Since m is constant, $|\mathcal{K}_n| \in \text{poly}(n)$. For $\widetilde{k}_n = (\widetilde{k}[1], \ldots, \widetilde{k}[m]) \in \mathcal{K}_n$, define $\widetilde{k}'_n = (\widetilde{k}'[1], \ldots, \widetilde{k}[m]')$ by $\widetilde{k}'[i] = \min\left\{0, v\widetilde{k}'[i] - c' \log(n) \cdot \ell \cdot \sqrt{v}\right\}$, for c' being the constant from Lemma 5.5.

By Theorem 7.1, the existence of G implies that of one-way functions. Hence, for any large enough n and $\widetilde{k}_n \in \mathcal{K}_n$, we have that $\mathsf{Com}(1^n, \widetilde{k}'_n)$ is:

1. correct,
2. computationally binding, and
3. statistically hiding if $\widetilde{k}_n = k_n$.

Consider the commitment scheme that on security parameter n, the parties invoke $\mathsf{Com}(1^n, \widetilde{k}'_n)$ in parallel, for all choices of $\widetilde{k}_n \in \mathcal{K}_n$, where the committed values used by the sender are $|\mathcal{K}_n|$-out-of-$|\mathcal{K}_n|$ shares of the value the sender wishes to commit to (i.e., their XOR is the committed value). By the above observation, the resulting commitment is the desired constant-round statistically hiding commitment.

Finally, it readily follows from the above proof that if the bound on the real entropy of G is invariant to the public parameter, then so is that of $\mathsf{G}^{\langle v \rangle}$, and the resulting commitment is receiver public-coin. □

7 One-Way Functions are Necessary for an Accessible Entropy Generator

In Section 4 we proved that the existence of one-way functions implies that of an inaccessible entropy generator. The following theorem states that the converse direction is also true.

Theorem 7.1. *Let G be an efficient block generator with real entropy $k(n)$. If G has accessible entropy at most $k(n) - 1/p(n)$ for some $p \in \text{poly}$, then one-way functions exist.*

Proof. The proof is by reduction. We assume that one-way functions do not exist, and show that this implies that G does not have a noticeable gap between its real and accessible entropy. For simplicity, we assume that G gets no public parameter.

Consider the efficient function that outputs the first i blocks of of G:

$$g(x, i) = \mathsf{G}(x)_{1,\ldots,i} \tag{31}$$

We assume for ease of notation that the seed length of G on security parameter n is n (i.e., $s(n) = n$). Hence, g is defined over $\{0,1\}^n \times [n]$.

Let m and ℓ be the block complexity and maximal block length of G respectively, and let $\alpha = 1/20m^3(n)p^2(n)(\ell(n) + n)$. Assuming one-way functions do not exist, by Lemma 2.18 there exists an efficient algorithm Inv that is an α-inverter for g (see Definition 2.16) on infinitely many n's. We assume without loss of generality that Inv either outputs a valid preimage of g, or $\perp$. Consider the following on-line generator $\widetilde{\mathsf{G}}$:

Algorithm 7.2 (Online generator $\widetilde{\mathsf{G}}(1^n)$)**.**

1. *Let* $\mathsf{Fail} = \mathsf{false}$ *and* $x_0 = 0^n$.
2. *For* $i = 1$ *to* $m(n)$*:*
   (a) *If* Fail*, set* $x_i = x_{i-1}$.
   Otherwise,
      i. *Let* $(x_i, \cdot) \leftarrow \mathsf{Inv}(\mathsf{G}(x_{i-1})_{1,\ldots,i-1})$.
      ii. *If* $x_i = \perp$*, set* $\mathsf{Fail} = \mathsf{true}$ *and* $x_i = x_{i-1}$.
   (b) *Output* $\mathsf{G}(x_i)_i$.

It is clear that $\widetilde{\mathsf{G}}$ is efficient and G-consistent. In the following we show that for infinitely many n's, the accessible entropy of $\widetilde{\mathsf{G}}$ is at least $k(n) - 1/p(n)$, contradicting the assumed bound on the accessible entropy of G.

Let $\mathcal{I} \subseteq \mathbb{N}$ be the infinite sequence of input lengths on which Inv is an α-inverter of g. Fix $n \in \mathcal{I}$ and omit it from the notation when clear from the context. Let $Y = (Y_1, \ldots, Y_m) = \mathsf{G}(U_n)$ and let $(R_1, \widetilde{Y}_1, \ldots, R_m, \widetilde{Y}_m) = \widetilde{T} = T_{\widetilde{\mathsf{G}}}(1^n)$. We first prove that $\mathrm{H}(\widetilde{Y} = (\widetilde{Y}_1, \ldots, \widetilde{Y}_m))$ is almost as high as the real entropy of G.

Claim 7.3. $\mathrm{H}(\widetilde{Y}) \geq \mathrm{H}(Y) - 1/3p$.

Proof. Since Inv is an α-inverter, a simple coupling argument yields that

$$\delta := \mathrm{SD}(Y, \widetilde{Y}) \leq 2m\alpha < 1/6pm\ell. \tag{32}$$

Hence, [31, Fact 3.3.9] yields that

$$\mathrm{H}(\widetilde{Y}) - \mathrm{H}(Y) \geq -\delta \cdot \log(\left|\mathrm{Supp}(\widetilde{Y} \cup Y)\right| - h_2(\delta), \tag{33}$$

for $h_2(x)$ being the Rényi entropy of the Boolean random variable taking the value 1 wp x. Since $\widetilde{\mathsf{G}}$ is non-failing, it holds that $\mathrm{Supp}(\widetilde{Y}) \subseteq \mathrm{Supp}(Y)$, and by assumption, $|\mathrm{Supp}(Y)| \leq 2^{m\ell}$. It follows that

$$\begin{aligned}
\mathrm{H}(\widetilde{Y}) - \mathrm{H}(Y) &\geq -\delta m\ell - h_2(\delta) \\
&\geq -\delta m\ell - 2\delta \\
&\geq -(m\ell + 2)/4pm\ell \\
&\geq -1/3p.
\end{aligned}$$

The second inequality holds since $h_2(\delta) < 2\delta$, and the third one since without loss of generality $\ell m \geq 2$. □

Recall that by definition,

$$\mathrm{AccH}_{\widetilde{\mathsf{G}}}(\widetilde{T}) = \mathsf{E}_{(r_1, y_1, \ldots) \leftarrow \widetilde{T}} \left[\sum_{i=1}^{m} \mathrm{H}_{\widetilde{Y}_i | R_{<i}}(y_i | r_{<i}) \right] \tag{34}$$

and that, by the chain rule,

$$\mathrm{H}(\widetilde{Y}) = \mathsf{E}_{(r_1,y_1,\ldots)\leftarrow\widetilde{T}}\left[\sum_{i=1}^{m}\mathrm{H}_{\widetilde{Y}_i|\widetilde{Y}_{<i}}(y_i|y_{<i})\right] \tag{35}$$

We complete the proof showing that with save but very small probability over the choice of $(r_1, y_1, \ldots) \leftarrow \widetilde{T}$, it holds that $\mathrm{H}_{\widetilde{Y}_i|R_{<i}}(y_i|r_{<i})$ is very close to $\mathrm{H}_{Y_i|Y_{<i}}(y_i|y_{<i})$ for every $j \in [m]$, and that the complementary event does not contribute much to the entropy of Y. We do that by focusing on the set of "good" transcripts $\mathcal{S} \subseteq \mathrm{Supp}(\widetilde{T})$. The set $\mathcal{S}$ contains all transcripts $\mathbf{t} = (r_1, y_1, \ldots, r_m, y_m)$ such that

1. $\mathsf{Fail}(\mathbf{t}) = \mathsf{false}$, for $\mathsf{Fail}(\mathbf{t})$ being the event for which the flag Fail is set to true in the execution of $\widetilde{\mathsf{G}}$ reflected in $\mathbf{t}$.
2. $\mathrm{H}_{\widetilde{Y}_i|R_{<i}}(y_i|r_{<i}) + 1/3mp \geq \mathrm{H}_{\widetilde{Y}_i|\widetilde{Y}_{<i}}(y_i|y_{<i})$, for every $i \in [m]$.

We first prove that a random transcript is likely to be in $\mathcal{S}$.

Claim 7.4. $\mathsf{Pr}\left[\widetilde{T} \notin \mathcal{S}\right] \leq 1/4p(m\ell + n)$.

Proof. Let $\mathcal{S}' = \left\{\mathbf{t} = (r_1, y_1, \ldots) \in \mathrm{Supp}(\widetilde{T})\colon \forall i \in [m]\colon \mathrm{SD}\left((x)_{x\xleftarrow{\mathrm{R}}g^{-1}(y_{\leq i})}, \mathsf{Inv}(y_{\leq i})\right) \leq \alpha \wedge \mathsf{Fail}(\mathbf{t}) = \mathsf{false}\right\}$. Since Inv is an α-inverter, a simple coupling argument yields that

$$\mathsf{Pr}\left[\widetilde{T} \notin \mathcal{S}'\right] \leq 2m\alpha \tag{36}$$

We conclude the proof by showing that the second property of $\mathcal{S}$ holds with high probability for a random transcript of $\mathcal{S}'$. Fix $\mathbf{t} = (r_1, y_1, \ldots, r_m, y_m) \in \mathcal{S}'$. By definition, it holds that

$$\mathsf{Pr}\left[\widetilde{Y}_i = y_i \mid R_{<i} = r_{<i}\right] \leq \mathsf{Pr}\left[\widetilde{Y}_i = y_i \mid \widetilde{Y}_{<i} = y_{<i}\right] + \alpha \tag{37}$$

for every $\in [m]$. In particular, if $\mathsf{Pr}\left[\widetilde{Y}_i = y_i \mid \widetilde{Y}_{<i} = y_{<i}\right] \geq 3pm\alpha$, then

$$\mathrm{H}_{\widetilde{Y}_i|R_{<i}}(y_i|r_{<i}) + 1/3mp \geq \mathrm{H}_{\widetilde{Y}_i|\widetilde{Y}_{<i}}(y_i|y_{<i}) \tag{38}$$

Thus, we should only care about transcripts for which $\mathsf{Pr}\left[\widetilde{Y}_i = y_i \mid \widetilde{Y}_{<i} = y_{<i}\right] < 3pm\alpha$ for some i.

Note that for every possible value of $r_{<i}$, there exists at most a *single* value $y_i^* = y_i^*(r_{<i})$ such that $\mathsf{Pr}\left[\widetilde{Y}_i = y_i^* \mid R_{<i} = r_{<i}\right] > \mathsf{Pr}\left[\widetilde{Y}_i = y_i^* \mid \widetilde{Y}_{<i} = y_{<i}\right]$ (i.e., the value $\widetilde{\mathsf{G}}$ outputs when Inv fails). Assuming $\mathrm{SD}\left((x)_{x\xleftarrow{\mathrm{R}}g^{-1}(y_{<i})}, \mathsf{Inv}(y_{<i})\right) \leq \alpha$, it holds that

$$\mathsf{Pr}\left[\mathsf{Inv}(y_{<i}) = y\right] \leq 3pm\alpha + \alpha < 1/5mp(m\ell + n) \tag{39}$$

for every value y. Hence,

$$\mathsf{Pr}_{\mathbf{t}=(r_1,y_1,\ldots,r_m,y_m)\leftarrow\widetilde{T}}\left[\mathbf{t} \in \mathcal{S} \wedge \exists i \in [m]\colon y_i = y_i^*(r_{<i})\right] \leq 1/5p(m\ell + n) \tag{40}$$

We conclude that $\mathsf{Pr}\left[\widetilde{T} \in \mathcal{S}\right] \geq \mathsf{Pr}\left[\widetilde{T} \in \mathcal{S}'\right] - 1/5p(m\ell + n) \geq 1 - 1/4p(m\ell + n)$. □

We now use Claim 7.4 to show that the expectation of the sample-entropy $\widetilde{Y}$ is almost intact when ignoring the contribution of transcripts not in $\mathcal{S}$. By the second part of Proposition 2.8,

$$\mathsf{Pr}_{\mathbf{t}=(r_1,y_1,\ldots)\leftarrow\widetilde{T}}\left[\mathrm{H}_{\widetilde{Y}}(y=(y_1,\ldots,y_m)>\ell m+n\right]\leq 2^{-n} \tag{41}$$

Hence, the first part of Lemma 2.9 yields that for some universal constant c,

$$\mathsf{E}_{\mathbf{t}=(r_1,y_1,\ldots)\leftarrow\widetilde{T}}\left[(\mathrm{H}_{\widetilde{Y}}(y=(y_1,\ldots,y_m)>\ell m+n)\cdot\mathrm{H}_{\widetilde{Y}}(y)\right]\leq 2^{-n}(m\ell+n+n+c)<2^{-n/2} \tag{42}$$

for large enough n. It follows that

$$\mathsf{E}_{\mathbf{t}=(r_1,y_1,\ldots)\leftarrow\widetilde{T}}\left[(\mathbf{t}\notin\mathcal{S})\cdot\mathrm{H}_{\widetilde{Y}}(y_1,\ldots,y_m)\right]\leq(\ell m+n)\cdot 1/4p(\ell m+n)+2^{-n/2}<1/3p.$$

Thus,

$$\mathsf{E}_{\mathbf{t}=(r_1,y_1,\ldots)\leftarrow\widetilde{T}}\left[(\mathbf{t}\in\mathcal{S})\cdot\mathrm{H}_{\widetilde{Y}}(y_1,\ldots,y_m)\right]\geq\mathrm{H}(\widetilde{Y})-1/3p \tag{43}$$

We conclude that

$$\begin{aligned}
\mathrm{AccH}_{\widetilde{\mathsf{G}}}(\widetilde{T}) &= \mathsf{E}_{\mathbf{t}=(r_1,y_1,\ldots)\leftarrow\widetilde{T}}\left[\sum_{i=1}^{m}\mathrm{H}_{\widetilde{Y}_i|R_{<i}}(y_i|r_{<i})\right]\\
&\geq \mathsf{E}_{\mathbf{t}=(r_1,y_1,\ldots)\leftarrow\widetilde{T}}\left[(\mathbf{t}\in\mathcal{S})\cdot(\sum_{i=1}^{m}\mathrm{H}_{\widetilde{Y}_i|R_{<i}}(y_i|r_{<i}))\right]\\
&\geq \mathsf{E}_{\mathbf{t}=(r_1,y_1,\ldots)\leftarrow\widetilde{T}}\left[(\mathbf{t}\in\mathcal{S})\cdot(\sum_{i=1}^{m}\mathrm{H}_{\widetilde{Y}_i|\widetilde{Y}_{<i}}(y_i|y_{<i})-1/3mp)\right]\\
&\geq \mathsf{E}_{\mathbf{t}=(r_1,y_1,\ldots)\leftarrow\widetilde{T}}\left[(\mathbf{t}\in\mathcal{S})\cdot\mathrm{H}_{\widetilde{Y}}(y_1,\ldots,y_m)\right]-1/3p\\
&\geq \mathrm{H}(\widetilde{Y})-1/3p-1/3p\\
&> \mathrm{H}(Y)-1/p.
\end{aligned}$$

□

Acknowledgements

We thank Rosario Gennaro, Oded Goldreich and Muthuramakrishnan Venkitasubramaniam for very helpful discussions.

References

[1] B. Barak, R. Shaltiel, and A. Wigderson. Computational analogues of entropy. In *RANDOM-APPROX*, 2003.

[2] I. Berman, I. Haitner, and A. Tentes. Coin flipping of any constant bias implies one-way functions. *Journal of the ACM*, 65(3):14, 2018. Preliminary version in *STOC '14*.

[3] N. Bitansky, I. Haitner, I. Komargodski, and E. Yogev. Distributional collision resistance beyond one-way function. In *Advances in Cryptology – EUROCRYPT 2019*, 2019.

[4] M. Blum and S. Micali. How to generate cryptographically strong sequences of pseudo random bits. In *Proceedings of the 23rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 112–117, 1982.

[5] J. L. Carter and M. N. Wegman. Universal classes of hash functions. *Journal of Computer and System Sciences*, 18(2):143–154, Apr. 1979.

[6] J. L. Carter and M. N. Wegman. New hash functions and their use in authentication and set equality. *Journal of Computer and System Sciences*, 1981.

[7] Y. Z. Ding, D. Harnik, A. Rosen, and R. Shaltiel. Constant-round oblivious transfer in the bounded storage model. In *Theory of Cryptography, First Theory of Cryptography Conference, TCC 2004*, pages 446–472, 2004.

[8] Y. Dodis, R. Ostrovsky, L. Reyzin, and A. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM Journal on Computing*, 38(1):97–139, 2008.

[9] S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.

[10] I. Haitner and O. Reingold. Statistically-hiding commitment from any one-way function. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing (STOC)*, 2007.

[11] I. Haitner and S. Vadhan. *The Many Entropies in One-Way Functions*, pages 159–217. 2017.

[12] I. Haitner, O. Horvitz, J. Katz, C.-Y. Koo, R. Morselli, and R. Shaltiel. Reducing complexity assumptions for statistically hiding commitment. *Journal of Cryptology*, 22(3):283–310, 2009.

[13] I. Haitner, M. Nguyen, S. J. Ong, O. Reingold, and S. Vadhan. Statistically hiding commitments and statistical zero-knowledge arguments from any one-way function. *SIAM Journal on Computing*, pages 1153–1218, 2009. Preliminary versions in *FOCS '06* and *STOC '07*.

[14] I. Haitner, O. Reingold, S. Vadhan, and H. Wee. Inaccessible entropy. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing (STOC)*, 2009.

[15] I. Haitner, T. Holenstein, O. Reingold, S. Vadhan, and H. Wee. Universal one-way hash functions via inaccessible entropy. In *Advances in Cryptology – EUROCRYPT 2010*, 2010.

[16] I. Haitner, D. Harnik, and O. Reingold. On the power of the randomized iterate. *SIAM Journal on Computing*, 40(6):1486–1528, 2011. Preliminary version in *Crypto'06*.

[17] I. Haitner, O. Reingold, and S. Vadhan. Efficiency improvements in constructing pseudorandom generators from one-way functions. *SIAM Journal on Computing*, 42(3):1405–1430, 2013. Special Issue on *STOC '10*.

[18] I. Haitner, J. J. Hoch, O. Reingold, and G. Segev. Finding collisions in interactive protocols. Tight lower bounds on the round and communication complexities of statistically hiding commitments. *SIAM Journal on Computing*, 44(1):193–242, 2015.

[19] J. Håstad, R. Impagliazzo, L. A. Levin, and M. Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999. Preliminary versions in *STOC'89* and *STOC'90*.

[20] J. Hästad, R. Pass, K. Pietrzak, and D. Wikström. An efficient parallel repetition theorem. In *Theory of Cryptography, Sixth Theory of Cryptography Conference, TCC 2010*, 2010.

[21] R. Impagliazzo. *Pseudo-random generators for cryptography and for randomized algorithms*. PhD thesis, University of California, Berkeley, 1992. http://cseweb.ucsd.edu/~russell/format.ps.

[22] R. Impagliazzo and M. Luby. One-way functions are essential for complexity based cryptography. In *Proceedings of the 30th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 230–235, 1989.

[23] R. Impagliazzo and D. Zuckerman. How to recycle random bits. In *Proceedings of the 30th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 248–253, 1989.

[24] R. Impagliazzo, L. A. Levin, and M. Luby. Pseudo-random generation from one-way functions. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing (STOC)*, pages 12–24. ACM Press, 1989.

[25] J. Katz and C. Koo. On constructing universal one-way hash functions from arbitrary one-way functions. Technical Report 2005/328, Cryptology ePrint Archive, 2005.

[26] M. Naor. Bit commitment using pseudorandomness. *Journal of Cryptology*, 4(2):151–158, 1991. Preliminary version in *CRYPTO'89*.

[27] M. Naor and M. Yung. Universal one-way hash functions and their cryptographic applications. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing (STOC)*, pages 33–43. ACM Press, 1989.

[28] J. Rompel. One-way functions are necessary and sufficient for secure signatures. In *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing (STOC)*, pages 387–394, 1990.

[29] C. Shannon. Communication theory of secrecy systems. *Bell System Technical Journal*, 28(4): 656–715, 1949.

[30] S. Vadhan and C. J. Zheng. Characterizing pseudoentropy and simplifying pseudorandom generator constructions. In *Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC)*, pages 817–836, 2012.

[31] S. P. Vadhan. *A Study of Statistical Zero-Knowledge Proofs*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 1999.

[32] G. Yang. *Cryptography and Randomness Extraction in the Multi-Stream Model*. PhD thesis, Tsinghua University, Beijing, China, 2015. http://eccc.hpi-web.de/static/books/Cryptography_and_Randomness_Extraction_in_the_Multi_Stream_Model.

[33] A. C. Yao. Theory and applications of trapdoor functions. In *FOCS*, pages 80–91, 1982.

A Maximal Accessible Entropy

In this section we formally define the *maximal (max)* accessible entropy of a generator and provide basic observations and tools to work with this measure. Working with accessible max-entropy raises some additional subtleties. In particular, to make the manipulation defined in Section 5 applicable to this measure, we need to strengthen the notion of accessible entropy, so that it takes into consideration "preprocessing randomness".

Definition A.1 (Online block-generator, preprocessing variant)**.** *Let n be a security parameter, and let $c = c(n)$ and $m = m(n)$. An m-block* online *generator is a function $\widetilde{\mathsf{G}}\colon \{0,1\}^c \times (\{0,1\}^v)^{m+1} \mapsto (\{0,1\}^*)^m$ for some $v = v(n)$, such that the i'th output block of $\widetilde{\mathsf{G}}$ is a function of (only) its first $i+1$ input blocks. We denote the* transcript *of $\widetilde{\mathsf{G}}$ over random input by $T_{\widetilde{\mathsf{G}}}(1^n) = (Z, R_0, R_1, Y_1, \ldots, R_m, Y_m)$, for $Z \in \{0,1\}^c$, $(R_0, R_1, \ldots, R_m) \xleftarrow{R} (\{0,1\}^v)^{m+1}$ and $(Y_1, \ldots, Y_m) = \widetilde{\mathsf{G}}(Z, R_0, R_1, \ldots, R_i)$.*

That is, unlike the definition given in Section 3.2, the generator's first block is a function of its first *two* random strings (i.e., r_0 and r_1). The role of the first string (r_0) is to allow the generator a (randomized) "preprocessing stage" before it starts computing the output blocks, and the accessible entropy of the generator will be measured with respect to a random choice of this *preprocessing randomness*. This preprocessing randomness becomes handy when bounding the accessible entropy of a generator constructed by manipulating (e.g., repetition) of another generator, as done in Appendix A.1.[19]

Definition A.2 (Accessible sample-entropy, preprocessing variant)**.** *Let n be a security parameter, and let $\widetilde{\mathsf{G}}$ be an online $m = m(n)$-block online generator. The* accessible sample-entropy of $\mathbf{t} = (z, r_0, r_1, y_1, \ldots, r_m, y_m) \in \mathrm{Supp}(Z, R_0, R_1, Y_1 \ldots, R_m, Y_m) = T_{\widetilde{\mathsf{G}}}(1^n)$ *is defined by*

$$\mathrm{AccH}^{\widetilde{\mathsf{G}},n}(\boldsymbol{t}) = \sum_{i=1}^{m} \mathrm{H}_{Y_i|Z,R_{<i}}(y_i|z, r_{<i}).$$

That is, the surprise in y_1, i.e., $\mathrm{H}_{Y_1|Z,R_{<1}}(y_1|z, r_{<1})$, is measured also with respect to the value of R_0. This might increase the sample-entropy of its output blocks when a non-typical value for r_0 is sampled.

The average accessible entropy of a generator with preprocessing is defined as in Definition 3.10 with respect to the above notion of sample-entropy. It is not hard to see that the two quantities are the same (for any generator). For accessible max-entropy, the preprocessing quantity defined next is more manipulation friendly, and we do not know that is equivalent to its non-preprocessing variant.

Definition A.3 (Max accessible entropy, preprocessing variant)**.** *A block generator G has* accessible max-entropy at most *k if for every efficient G-consistent, online generator $\widetilde{\mathsf{G}}$ and all large enough n,*

$$\mathsf{Pr}_{\boldsymbol{t} \xleftarrow{R} T_{\widetilde{\mathsf{G}}}(1^n)}[\mathrm{AccH}^{\widetilde{\mathsf{G}}}(\boldsymbol{t}) > k] = \mathrm{neg}(n),$$

for every such $\widetilde{\mathsf{G}}$.

[19] When considering *nonuniform* generators, as done in [11], there is no need for the preprocessing randomness, since the generator can fix the best choice for this part of its random coins.

We first note that the accessible max-entropy of a generator indeed bounds its accessible entropy.

Lemma A.4. *Let* G *be an efficient block generator of accessible max-entropy* k*. Then its accessible entropy is at most* $k(n) + 1/p(n)$*, for any* $p \in \text{poly}$*.*

Proof. Fix an efficient G-consistent, online generator $\widetilde{\mathsf{G}}$ and $p \in \text{poly}$. Let m and ℓ be the block complexity and maximal block length of G. By assumption, for large enough n it holds that

$$\mathsf{Pr}_{\mathbf{t} \xleftarrow{R} T_{\widetilde{\mathsf{G}}}(1^n)}[\text{AccH}^{\widetilde{\mathsf{G}}}(\mathbf{t}) > k(n)] \leq \varepsilon(n)$$

for $\varepsilon(n) = 1/p(n)\ell(n)m(n)n$. Since $\varepsilon(m\ell - \log 1/\varepsilon) \leq 1/p(n)$, a similar proof to that given in Lemma 3.8 yields that $\mathsf{E}\left[\text{AccH}^{\widetilde{\mathsf{G}}}_{\widetilde{\mathsf{G}}}(T_{\widetilde{\mathsf{G}}}(1^n))\right] \leq k(n) + 1/p(n)$. □

The following theorem is the accessible max-entropy entropy variant of Theorem 4.2, stating that the accessible max-entropy of the one-way function generator described in Construction 4.1 is $n - \omega(\log n)$.

Theorem A.5 (Max Inaccessible entropy generators from one-way functions)**.** *If* $f\colon \{0,1\}^n \mapsto \{0,1\}^n$ *is one-way, then the efficient block-generator* $\mathsf{G} = \mathsf{G}^f$ *defined in Construction 4.1 has accessible* max*-entropy* $n - \omega(\log n)$*.*

By Lemma A.4, Theorem A.5 implies Theorem 4.2, but working with max-entropy its proof is significantly more complicated.

Proof of Theorem A.5. Suppose Theorem A.5 does not hold, and let $\widetilde{\mathsf{G}}$ be an efficient, G-consistent online block-generator such that

$$\mathsf{Pr}_{\mathbf{t} \xleftarrow{R} T_{\widetilde{\mathsf{G}}}(1^n)}\left[\text{AccH}^{\widetilde{\mathsf{G}}}_{\widetilde{\mathsf{G}}}(\mathbf{t}) > n - c \cdot \log n\right] > \varepsilon(n) \tag{44}$$

for some $\varepsilon(n) = 1/\text{poly}(n)$, and infinitely many n's. In the following, we fix $n \in \mathbb{N}$ for which the above equation holds, and omit it from the notation when its value is clear from the context. Let $m = n/\log n + 1$ and let v be a bound on the number of coins used by $\widetilde{\mathsf{G}}$ in each round. The inverter Inv for f is defined as follows:

Algorithm A.6 (Inverter Inv for f from the accessible max-entropy generator $\widetilde{\mathsf{G}}$)**.**

Input: $z \in \{0,1\}^n$

Operation:

1. *For* $i = 1$ *to* $m - 1$*:*
   (a) *Sample* $r_i \xleftarrow{R} \{0,1\}^v$ *and let* $y_i = \widetilde{\mathsf{G}}(r_1, \ldots, r_i)_i$*.*
   (b) *If* $y_{i \cdot \log n + 1, \ldots, (i+1) \cdot \log n} = z_i$*, move to next value of* i*.*
   (c) *Abort after* n^3/ε *failed attempts for sampling a good* r_i*.*
2. *Sample* $r_m \xleftarrow{R} \{0,1\}^v$ *and output* $\widetilde{\mathsf{G}}(r_1, \ldots, r_m)_m$*.*

It is clear that Inv runs in polynomial time, so we will finish the proof by showing that

$$\mathsf{Pr}_{y \xleftarrow{R} f(U_n)}\left[\mathsf{Inv}(y) \in f^{-1}(y)\right] \geq \varepsilon^2/16n.$$

We prove the above by relating the transcript distribution induced by the standalone execution of $\widetilde{\mathsf{G}}(1^n)$ to that induced by the execution of $\widetilde{\mathsf{G}}$ embedded (emulated) in $\mathsf{Inv}(f(U_n))$. In more detail, we show that high-accessible-entropy transcripts with respect to the standalone execution of G, i.e., $\mathrm{AccH}^{\widetilde{\mathsf{G}}}(\mathbf{t}) > n - c \cdot \log n$, are produced with not much smaller probability also in the embedded execution. Since whenever Inv does not abort it inverts y, it follows that the success probability of Inv is lower bounded by the probability that $\widetilde{\mathsf{G}}(1^n)$ outputs a high-accessible-entropy transcript, and thus is non-negligible.

For intuition about why the above statement about high-accessible-entropy transcripts is true, consider the case of a one-way *permutation* f. By definition, high-accessible-entropy transcripts in the standalone execution of $\widetilde{\mathsf{G}}$ are produced with probability at most $\mathrm{poly}(n)/2^n$. On the other hand, the probability that a "typical" transcript is produced by the emulated execution of $\widetilde{\mathsf{G}}$ is about 2^{-n} : the probability that a random output of f equals the transcript's first n output blocks.

We now formally prove the above for arbitrary one-way functions.

Standalone execution $\widetilde{\mathsf{G}}(1^n)$. Let $\widetilde{T} = T_{\widetilde{\mathsf{G}}}$, and recall that $\widetilde{T} = (\widetilde{R}_1, \widetilde{Y}_1, \ldots, \widetilde{R}_m, \widetilde{Y}_m)$ is associated with a random execution of $\widetilde{\mathsf{G}}$ on security parameter n by

- $\widetilde{R}_i$ – the random coins of $\widetilde{\mathsf{G}}$ in the i'th round, and
- $\widetilde{Y}_i$ – $\widetilde{\mathsf{G}}$'s i'th output block.

Recall that for $\mathbf{t} = (r_1, y_1, \ldots, r_m, y_m) \in \mathrm{Supp}(\widetilde{T})$, we have defined

$$\mathrm{AccH}^{\widetilde{\mathsf{G}}}(\mathbf{t}) := \sum_{i \in [m]} \mathrm{H}_{Y_j | R_{<j}}(y_j | r_{<j}).$$

Compute

$$\begin{aligned}
\mathsf{Pr}_{\widetilde{T}}[\mathbf{t}] &= \prod_{i=1}^{m} \mathsf{Pr}_{\widetilde{Y}_i | \widetilde{R}_{<i}}[y_i | r_{<i}] \cdot \mathsf{Pr}_{\widetilde{R}_i | \widetilde{R}_{<i}, \widetilde{Y}_i}[r_i | r_{<i}, y_i] \qquad (45)\\
&= 2^{-\sum_{i=1}^{m} \mathrm{H}_{\widetilde{Y}_i | \widetilde{R}_{<i}}(y_i | r_{<i})} \cdot \prod_{i=1}^{m} \mathsf{Pr}_{\widetilde{R}_i | \widetilde{R}_{<i}, \widetilde{Y}_i}[r_i | r_{<i}, y_i]\\
&= 2^{-\mathrm{AccH}^{\widetilde{\mathsf{G}}}(\mathbf{t})} \cdot R(\mathbf{t})
\end{aligned}$$

for

$$R(\mathbf{t}) := \prod_{i=1}^{m} \mathsf{Pr}_{\widetilde{R}_i | \widetilde{R}_{<i}, \widetilde{Y}_i}[r_i | r_{<i}, y_i] \qquad (46)$$

Execution embedded in $\mathsf{Inv}^{\widetilde{\mathsf{G}}}(f(U_n))$. Let $\widehat{T} = (\widehat{R}_1, \widehat{Y}_1, \ldots, \widehat{R}_m, \widehat{Y}_m)$ denote the value of $\widetilde{\mathsf{G}}$'s coins and output blocks, sampled in Step 1 of a random execution of the *unbounded* version of Inv (i.e., Step 1c is removed) on input $Z = (Z_1, \ldots, Z_{m-1}) = f(U_n)$. (This unboundedness change is only an intermediate step in the proof that does not significantly change the inversion probability of Inv, as shown below.)

Since $\widetilde{\mathsf{G}}$ is G-consistent, it holds that $(y_1, \ldots, y_{m-1}) \in \mathrm{Supp}(f(U_n))$ for every $(r_1, y_1, \ldots, r_m, y_m) \in \mathrm{Supp}(\widetilde{T})$. It follows that every $\mathbf{t} \in \mathrm{Supp}(\widetilde{T})$ can be "produced" by the unbounded version of Inv, and therefore $\mathrm{Supp}(\widetilde{T}) \subseteq \mathrm{Supp}(\widehat{T})$. For $\mathbf{t} = (r_1, y_1, \ldots, r_m, y_m) \in \mathrm{Supp}(\widetilde{T})$, compute

$$\begin{aligned}
\mathsf{Pr}_{\widehat{T}}[\mathbf{t}] &= \mathsf{Pr}_{\widehat{R}_0}[r_0] \cdot \prod_{i=1}^{m} \mathsf{Pr}_{\widehat{Y}_i|\widehat{R}_{<i}}[y_i|r_{<i}] \cdot \mathsf{Pr}_{\widehat{R}_i|\widehat{R}_{<i},\widehat{Y}_i}[r_i|r_{<i}, y_i] && (47)\\
&= \left(\prod_{i=1}^{m-1} \mathsf{Pr}_{Z_i|\widehat{Y}_{<i}}[y_i|y_{<i}] \cdot \mathsf{Pr}_{\widehat{Y}_i|\widehat{R}_{<i},Z_i}[y_i|r_{<i}, y_i]\right) \cdot \mathsf{Pr}_{\widehat{Y}_m|\widehat{R}_{<m}}[y_m|r_{<m}] \cdot \prod_{i=0}^{m} \mathsf{Pr}_{\widehat{R}_i|\widehat{R}_{<i},\widehat{Y}_i}[r_i|r_{<i}, y_i]\\
&= \left(\prod_{i=1}^{m-1} \mathsf{Pr}_{Z_i|\widehat{Y}_{<i}}[y_i|y_{<i}] \cdot 1\right) \cdot \mathsf{Pr}_{\widehat{Y}_m|\widehat{R}_{<m}}[y_m|r_{<m}] \cdot \prod_{i=0}^{m} \mathsf{Pr}_{\widehat{R}_i|\widehat{R}_{<i},\widehat{Y}_i}[r_i|r_{<i}, y_i] && (48)\\
&= \mathsf{Pr}_{f(U_n)}[y_{<m}] \cdot \mathsf{Pr}_{\widehat{Y}_m|\widehat{R}_{<m}}[y_m|r_{<m}] \cdot \prod_{i=0}^{m} \mathsf{Pr}_{\widehat{R}_i|\widehat{R}_{<i},\widehat{Y}_i}[r_i|r_{<i}, y_i]\\
&= \mathsf{Pr}_{f(U_n)}[y_{<m}] \cdot \mathsf{Pr}_{\widetilde{Y}_m|\widetilde{R}_{<m}}[y_m|r_{<m}] \cdot \prod_{i=0}^{m} \mathsf{Pr}_{\widehat{R}_i|\widehat{R}_{<i},\widehat{Y}_i}[r_i|r_{<i}, y_i] && (49)\\
&= \mathsf{Pr}_{f(U_n)}[y_{<m}] \cdot \mathsf{Pr}_{\widetilde{Y}_m|\widetilde{R}_{<m}}[y_m|r_{<m}] \cdot \prod_{i=0}^{m} \mathsf{Pr}_{\widetilde{R}_i|\widetilde{R}_{<i},\widetilde{Y}_i}[r_i|r_{<i}, y_i] && (50)\\
&= \mathsf{Pr}_{f(U_n)}[y_{<m}] \cdot \mathsf{Pr}_{\widetilde{Y}_m|\widetilde{R}_{<m}}[y_m|r_{<m}] \cdot R(\mathbf{t}),
\end{aligned}$$

where again, we let $\widehat{Y}_0$ and y_0 stand for the empty strings.

Equation (48) holds since $\mathbf{t} \in \mathrm{Supp}(\widetilde{T})$ and Inv is unbounded. Equation (49) holds since in both $\widetilde{T}$ and in $\widehat{T}$, the last output block has the same distribution conditioned on all but the last randomness block. Equation (50) holds since when conditioning on the value of the i'th output block, the randomness used to create this block is distributed the same in $\widetilde{T}$ and in $\widehat{T}$.

Relating the two distributions. Combining Equations (45) and (47) yields that, for $\mathbf{t} = (r_1, y_1, \ldots, r_m, y_m) \in \mathrm{Supp}(\widetilde{T})$, it holds that

$$\mathsf{Pr}_{\widehat{T}}[\mathbf{t}] = \mathsf{Pr}_{\widetilde{T}}[\mathbf{t}] \cdot \left(\mathsf{Pr}_{f(U_n)}[y_{<m}] \cdot \mathsf{Pr}_{\widetilde{Y}_m|\widetilde{R}_{<m}}[y_m|r_{<m}] \cdot 2^{\mathrm{AccH}^{\widetilde{\mathsf{G}}}(\mathbf{t})}\right) \tag{51}$$

In particular, if $\mathrm{AccH}^{\widetilde{\mathsf{G}}}(\mathbf{t}) \geq n - c\log n$, then

$$\begin{aligned}
\mathsf{Pr}_{\widehat{T}}[\mathbf{t}] &\geq \mathsf{Pr}_{\widetilde{T}}[\mathbf{t}] \cdot \frac{2^n \cdot \mathsf{Pr}_{f(U_n)}[y_{<m}]}{n^c} \cdot \mathsf{Pr}_{\widetilde{Y}_m|\widetilde{R}_{<m}}[y_m|r_{<m}] && (52)\\
&= \mathsf{Pr}_{\widetilde{T}}[\mathbf{t}] \cdot \frac{\left|f^{-1}(y_{<m})\right|}{n^c} \cdot \mathsf{Pr}_{\widetilde{Y}_m|\widetilde{R}_{<m}}[y_m|r_{<m}].
\end{aligned}$$

If it is also the case that $\mathrm{H}_{\widetilde{Y}_m|\widetilde{R}_{<m}}(y_m|r_{<m}) \leq \log\left|f^{-1}(y_{<m})\right| + k$ for some $k > 0$, then

$$\mathsf{Pr}_{\widehat{T}}[\mathbf{t}] \geq \mathsf{Pr}_{\widetilde{T}}[\mathbf{t}] \cdot \frac{\left|f^{-1}(y_{<m})\right|}{n^c} \cdot \frac{2^{-k}}{|f^{-1}(y_{<m})|} = \frac{\mathsf{Pr}_{\widetilde{T}}[\mathbf{t}]}{2^k n^c} \tag{53}$$

Lower bounding the inversion probability of Inv. We conclude the proof by showing that Equation (53) implies the existence of a large set of transcripts that (the bounded version of) Inv performs well upon.

Let $\mathcal{S}$ denote the set of transcripts $\mathbf{t} = (r_1, y_1, \ldots, r_m, y_m) \in \mathrm{Supp}(\widetilde{T})$ with

1. $\mathrm{Acc}\mathrm{H}_{\widetilde{\mathsf{G}}}(\mathbf{t}) \geq n - c\log n$,
2. $\mathrm{H}_{\widetilde{Y}_m|\widetilde{R}_{<m}}(y_m|r_{<m}) \leq \log\left|f^{-1}(y_{<m})\right| + \log(4/\varepsilon)$, and
3. $\mathrm{H}_{\widetilde{Y}_i|\widetilde{Y}_{<i}}(y_i|y_{<i}) \leq \log(4n^2/\varepsilon)$ for all $i \in [m-1]$.

The first two properties will allow us to use Equations (52) and (53) to argue that, if $\mathcal{S}$ happens with significant probability with respect to $\widetilde{T}$, then this holds also with respect to $\widehat{T}$. The last property will allow us to show that this also holds with respect to the bounded version of Inv. We start by showing that $\mathcal{S}$ happens with significant probability with respect to $\widetilde{T}$, then show that this holds also with respect to $\widehat{T}$, and finally use it to lowerbound the success probability of Inv.

By Lemma 2.1,

$$\mathsf{Pr}_{(r_1,y_1,\ldots,r_m,y_m)\xleftarrow{\mathrm{R}}\widetilde{T}}\left[\mathrm{H}_{\widetilde{Y}_m|\widetilde{R}_{<m}}(y_m|r_{<m}) > \log\left|f^{-1}(y_{<m})\right| + k\right] < 2^{-k} \tag{54}$$

for any $k > 0$. Since $\left|\mathrm{Supp}(\widetilde{Y}_i)\right| = n$ for all $i \in [m-1]$, it follows that

$$\mathsf{Pr}_{(y_1,\ldots,y_m)\xleftarrow{\mathrm{R}}(\widetilde{Y}_1,\ldots,\widetilde{Y}_m)}\left[\exists i \in [m-1]\colon \mathrm{H}_{\widetilde{Y}_i|\widetilde{Y}_{<i}}(y_i|y_{<i}) > v\right] < (m-1)\cdot n \cdot 2^{-v} \tag{55}$$

for any $v > 0$.

Applying Equations (54) and (55) with $k = \log(4/\varepsilon)$ and $v = \log(4mn/\varepsilon)$, respectively, and recalling that, by assumption, $\mathsf{Pr}_{\mathbf{t}\xleftarrow{\mathrm{R}}\widetilde{T}}\left[\mathrm{Acc}\mathrm{H}_{\widetilde{\mathsf{G}}}(\mathbf{t}) \geq n - c\log n\right] \geq \varepsilon$, yields that

$$\mathsf{Pr}_{\widetilde{T}}[\mathcal{S}] \geq \varepsilon - \frac{\varepsilon}{4} - \frac{\varepsilon}{4} = \varepsilon/2 \tag{56}$$

By Equation (53) and the first two properties of $\mathcal{S}$, we have that

$$\mathsf{Pr}_{\widehat{T}}[\mathcal{S}] \geq \frac{\varepsilon}{4n^c} \cdot \mathsf{Pr}_{\widetilde{T}}[\mathcal{S}] \geq \frac{\varepsilon^2}{8n^c} \tag{57}$$

Finally, let $\widehat{T}'$ denote the final value of $\widetilde{\mathsf{G}}$'s coins and output blocks, induced by the *bounded* version of Inv (set to $\perp$ if Inv aborts). The third property of $\mathcal{S}$ yields that

$$\mathsf{Pr}_{\widehat{T}'}[\mathbf{t}] \geq \mathsf{Pr}_{\widehat{T}}[\mathbf{t}] \cdot \left(1 - (m-1)\cdot\left(1 - \tfrac{\varepsilon}{4n^2}\right)^{n^3/\varepsilon}\right) \geq \mathsf{Pr}_{\widehat{T}}[\mathbf{t}] \cdot (1 - O(m\cdot 2^{-n})) \geq \mathsf{Pr}_{\widehat{T}}[\mathbf{t}]/2 \tag{58}$$

for every $\mathbf{t} \in \mathcal{S}$. We conclude that

$$\begin{aligned}\mathsf{Pr}_{z \xleftarrow{R} f(U_n)}\left[\mathsf{Inv}(z) \in f^{-1}(z)\right] &= \mathsf{Pr}_{z \xleftarrow{R} f(U_n)}\left[\mathsf{Inv}(z) \text{ does not abort}\right] \\ &\geq \mathsf{Pr}_{\widehat{T}'}\left[\mathcal{S}\right] \\ &\geq \frac{1}{2} \cdot \mathsf{Pr}_{\widehat{T}}[\mathcal{S}] \\ &\geq \frac{\varepsilon^2}{16n^c}.\end{aligned}$$

□

A.1 Manipulating Accessible Max-Entropy

In this section we analyze the effect of the tools introduced in Section 5 on the accessible max-entropy of the generator (rather than on average accessible entropy). The following statements and proofs are similar to those in Section 5, but are somewhat more complicated due to the more complicated nature of accessible max-entropy, and the preprocessing string of the online generator is critical to these proofs.

A.1.1 Truncated Sequential Repetition

Lemma A.7. *For security parameter n, let $m = m(n)$ be a power of 2, let $s = s(n)$, let G be an efficient m-block generator over $\{0,1\}^s$, and let $w = w(n)$ be a polynomially computable and bounded integer function. Then $\mathsf{G}^{[w]}$ defined according to Definition 5.1 is an efficient[20] $((w-1)\cdot m)$-block generator such that the following holds: if G has accessible max-entropy at most $k_{\text{ACC}} = k_{\text{ACC}}(n)$, then $\mathsf{G}^{[w]}$ has accessible max-entropy at most*

$$k'_{\text{ACC}} := (w-2)\cdot k_{\text{ACC}} + 2 \cdot \mathrm{H}_0(\mathsf{G}(U_s)) + \log(m) + d$$

for any $d = d(n) \in \omega(\log n)$.

Roughly, each of the $(w-2)$ non-truncated executions of G embedded in $\mathsf{G}^{[w]}$ contributes its accessible entropy to the overall accessible entropy of $\mathsf{G}^{[w]}$. As in the average accessible entropy case, we pay the max-entropy of the two truncated executions of G embedded in $\mathsf{G}^{[w]}$. Working with the less friendly measure of accessible max-entropy costs us an additional super-logarithmic loss d, a price we do not pay when working with average accessible entropy.

Proof. To avoid notational clutter, let $\mathbb{G} = \mathsf{G}^{[w]}$. Let $\widetilde{\mathbb{G}}$ be an efficient $\mathbb{G}$ $(= \mathsf{G}^{[w]})$-consistent generator, and let

$$\varepsilon = \varepsilon(n) = \mathsf{Pr}_{\mathbf{t} \xleftarrow{R} \widetilde{\mathbb{T}}}\left[\mathrm{AccH}^{\widetilde{\mathbb{G}}}(\mathbf{t}) > k'_{\text{ACC}}\right] \tag{59}$$

for $\widetilde{\mathbb{T}} = T_{\widetilde{\mathbb{G}}}(1^n)$. Our goal is to show that ε is negligible in n. We do that by showing that a random sub-transcript of $\widetilde{\mathbb{T}}$ contributes more than k_{ACC} bits of accessible entropy if the overall accessible

[20] Since m is a power of 2, standard techniques can be applied to change the input domain of $\mathsf{G}^{[w]}$ to $\{0,1\}^{s'}$ for some polynomial-bounded and polynomial-time computable s', making it an efficient block-generator according to Definition 3.1.

entropy of $\widetilde{\mathbb{T}}$ is more than k'_{ACC}. We then use this observation to construct a cheating generator for G that achieves accessible entropy greater than k_{ACC} with probability that is negligibly close to ε.

Let $(\mathbf{Z}, R_0, R_1, Y_1, \ldots, R_{\tilde{m}}, Y_{\tilde{m}}) = \widetilde{\mathbb{T}}$ and let J be the first part of Y_1 (recall that Y_1 is of the form $(j, \cdot)$). Fix $j \in [m]$, and let $(\mathbf{Z}, R_0^j, R_1^j, Y_1^j, \ldots, R_{\tilde{m}}^j, Y_{\tilde{m}}^j) = \widetilde{\mathbb{T}}^j = \widetilde{\mathbb{T}}|_{J=j}$. Let $\mathcal{I} = \mathcal{I}(j)$ be the indices of the output blocks coming from the truncated executions of G in $\mathbb{G}$ (i.e., $\mathcal{I} = \{1, \ldots, m+1-j\} \cup \{\tilde{m}+2-j, \ldots, \tilde{m}\}$).

Our first step is to show that these blocks do not contribute much more entropy than the max-entropy of $\mathsf{G}(U_n)$. Indeed, by Proposition 2.8, letting $\boldsymbol{X} = (\mathbf{Z}, R_0^j, Y_1^j, R_1^j, \ldots, Y_{\tilde{m}}^j, R_{\tilde{m}}^j)$ and $\mathcal{J}$ being the indices of the blocks of $\mathcal{I}$ in $\boldsymbol{X}$, it holds that

$$\mathsf{Pr}_{\mathbf{t}=(\boldsymbol{z},r_0,r_1,y_1,\ldots,r_{\tilde{m}},y_{\tilde{m}}) \xleftarrow{R} \widetilde{\mathbb{T}}^j} \left[\sum_{i \in \mathcal{I}} \mathrm{H}_{Y_i^j | \mathbf{Z}, R_{<i}^j}(y_i | \boldsymbol{z}, r_{<i}) > 2 \cdot \mathrm{H}_0(\mathsf{G}(U_s)) + d/2 \right] \leq 2 \cdot 2^{-d/2} = \mathrm{neg}(n) \tag{60}$$

Namely, with save but negligible probability, the blocks that relate to the truncated executions of G in $\mathbb{G}$ do not contribute much more than their support size to the overall accessible entropy.

Our next step is to remove the conditioning on $J = j$ (that we have introduced to have the indices of interest fixed, which enabled us to use Proposition 2.8). By Lemma 2.1, for any $\boldsymbol{z} \in \mathrm{Supp}(\mathbf{Z})$ and $r_0 \in \mathrm{Supp}(R_0)$ it holds that

$$\mathsf{Pr}_{j \xleftarrow{R} J | \mathbf{Z}=\boldsymbol{z}, R_0=r_0} \left[\mathrm{H}_{J|\mathbf{Z},R_0}(j | \boldsymbol{z}, r_0) > \log(m) + d/2 \right] \leq 2^{-d/2} = \mathrm{neg}(n) \tag{61}$$

Let $(\boldsymbol{z}, r_0, r_1, y_1 = (j, \cdot), \ldots, r_{\tilde{m}}, y_{\tilde{m}}) \in \mathrm{Supp}(\widetilde{\mathbb{T}})$. For $i > 1$, it holds that $\mathrm{H}_{Y_i|\mathbf{Z},R_{<i}}(y_i|\boldsymbol{z}, r_{<i}) = \mathrm{H}_{Y_i^j|\mathbf{Z},R_{<i}^j}(y_i|\boldsymbol{z}, r_{<i})$, whereas for $i = 1$, it holds that $\mathrm{H}_{Y_1|\mathbf{Z},R_0}(y_1|\boldsymbol{z}, r_0) = \mathrm{H}_{J|\mathbf{Z},R_0}(j|\boldsymbol{z}, r_0) + \mathrm{H}_{Y_1^j|\mathbf{Z},R_0}(y_1|\boldsymbol{z}, r_0)$. Hence, by Equations (60) and (61) it holds that

$$\mathsf{Pr}_{\mathbf{t}=(\boldsymbol{z},r_0,r_1,y_1,\ldots,r_{\tilde{m}},y_{\tilde{m}}) \xleftarrow{R} \widetilde{\mathbb{T}}} \left[\sum_{i \in [\tilde{m}] \setminus \mathcal{I}(J)} \mathrm{H}_{Y_i|\mathbf{Z},R_{<i}}(y_i|\boldsymbol{z}, r_{<i}) > (w-2) \cdot k_{\text{ACC}} \right] \geq \varepsilon - \mathrm{neg}(n) \tag{62}$$

Let $\mathbb{F}(j) = \{km + 2 - j \colon k \in [w-2]\}$, i.e., the indices of the first blocks of the non-truncated executions of G in $\mathbb{G}$, when the first block of $\mathbb{G}$ is $(j, \cdot)$. It follows that

$$\mathsf{Pr}_{\mathbf{t}=(\boldsymbol{z},r_0,r_1,y_1=(j,\cdot),\ldots,r_{\tilde{m}},y_{\tilde{m}}) \xleftarrow{R} \widetilde{\mathbb{T}}; f \xleftarrow{R} \mathbb{F}(j)} \left[\sum_{i=f}^{f+m-1} \mathrm{H}_{Y_i|\mathbf{Z},R_{<i}}(y_i|\boldsymbol{z}, r_{<i}) > k_{\text{ACC}} \right] \geq (\varepsilon - \mathrm{neg}(n))/w \tag{63}$$

Consider the following efficient G-consistent generator.

Algorithm A.8 (Generator $\widetilde{\mathsf{G}}$)**.**

Input: public parameter $z \in \{0,1\}^c$.

Operation:

1. *Sample* $v \xleftarrow{R} \{2, \ldots, w-1\}$. *Let* $\boldsymbol{z} = (z_1, \ldots, z_w)$ *for* $z_v = z$, *and* z_i, *for* $i \neq v$, *sampled uniformly in* $\{0,1\}^c$.

   We will refer to the part of $\boldsymbol{z}$ *sampled by the generator as* $\boldsymbol{z}_{-v}$, *and (abusing notation) assume* $\boldsymbol{z} = (z, \boldsymbol{z}_{-v})$.

2. *Start a random execution of* $\widetilde{\mathbb{G}}(\boldsymbol{z})$. *After* $\widetilde{\mathbb{G}}$ *locally outputs its first block* $(j,\cdot)$, *continue the execution of* $\widetilde{\mathbb{G}}$ *while outputting, block by block, the output blocks of* $\widetilde{\mathbb{G}}$ *indexed by* $\{f=(v-1)m+2-j,\ldots,f+m-1\}$.

It is clear that $\widetilde{\mathsf{G}}$ is indeed an efficient G-consistent generator. We will show that the accessible entropy of $\widetilde{\mathsf{G}}$ violates the assumed bound on the accessible entropy of G.

Let $(Z,R'_0=(R_0,F,\mathbf{Z}_{-F}),R_1,Y_1,\ldots,R_m,Y_m)=T_{\widetilde{\mathsf{G}}}$ be the transcript of $\widetilde{\mathsf{G}}(Z)$. It is easy to verify that

$$\mathrm{H}_{Y_i|Z,R_{<i}}(y_i|z,r'_{<i})=\mathrm{H}_{Y_{f+i}|\mathbf{Z},R_{<f+i}}(y_i|\boldsymbol{z}=(z,\boldsymbol{z}_{-f}),r'_{<i}=(r'_0,r_1,\ldots,r_{<i})) \tag{64}$$

for every $(z,r'_0=(r_0,f,\boldsymbol{z}_{-f}),r_1,y_1,\ldots,r_m,y_m)\in\mathrm{Supp}(T_{\widetilde{\mathsf{G}}})$ and $1<i\leq m$. Thus, Equation (63) yields that

$$\mathsf{Pr}_{\mathbf{t}\xleftarrow{R}T_{\widetilde{\mathsf{G}}}}\left[\mathrm{AccH}_{\widetilde{\mathsf{G}}}(\mathbf{t})>k_{\mathrm{ACC}}\right]>(\varepsilon-\mathrm{neg}(n))/w.$$

The assumption about the inaccessible entropy of G yields that ε is a negligible function of n, and the proof of the lemma follows. □

A.1.2 Direct Product

Lemma A.9. *For security parameter* n, *let* $m=m(n)$, *let* $v=v(n)$ *be polynomial-time computable and bounded integer functions, and let* G *be an efficient*[21] m*-block generator. Then* $G^{\langle v\rangle}$, *defined according to Definition 5.4, is an efficient* m*-block generator such that the following holds: if* G *has accessible max-entropy at most* $k_{\mathrm{ACC}}=k_{\mathrm{ACC}}(n)$, *then* $G^{\langle v\rangle}$ *has accessible max-entropy at most* $k'_{\mathrm{ACC}}(n)=v\cdot k_{\mathrm{ACC}}+d\cdot m$, *for any* $d=d(n)\in\omega(\log n)$.

As in the truncated sequential repetition Lemma A.7, working with the less friendly measure of accessible max-entropy costs us an additional super-logarithmic loss , a price we do not pay when working with average accessible entropy.

Proof. Let $\mathbb{G}=G^{\langle v\rangle}$, let $\widetilde{\mathbb{G}}$ be an efficient $\mathbb{G}$-consistent generator, and let

$$\varepsilon=\varepsilon(n)=\mathsf{Pr}_{\mathbf{t}\xleftarrow{R}\widetilde{\mathbb{T}}}\left[\mathrm{AccH}_{\widetilde{\mathbb{G}}}(\mathbf{t})>k'_{\mathrm{ACC}}\right] \tag{65}$$

for $\widetilde{\mathbb{T}}=T_{\widetilde{\mathbb{G}}}(1^n)$. Our goal is to show that ε is negligible in n.

Let $(\mathbf{Z},R_0,R_1,Y_1,\ldots,R_m,Y_m)=\widetilde{\mathbb{T}}$. Recall that for $\mathbf{t}=(\boldsymbol{z},r_0,r_1,y_1,\ldots,r_m,y_m)\in\mathrm{Supp}(\widetilde{\mathbb{T}})$, we have defined

$$\mathrm{AccH}_{\widetilde{\mathbb{G}}}(\mathbf{t})=\sum_{i\in[m]}\mathrm{H}_{Y_i|\mathbf{Z}.R_{<i}}(y_i\mid\boldsymbol{z},r_{<i}) \tag{66}$$

[21] Again, standard techniques can be applied to change the input domain of G to $\{0,1\}^{s'(n)}$ for some polynomial-bounded and polynomial-time computable s', making it an efficient block-generator according to Definition 3.1.

Since $\widetilde{\mathbb{G}}$ is $\mathbb{G}$-consistent, each Y_i is of the form $(Y_{i,1},\ldots,Y_{i,v})$. Hence Lemma 2.11, taking $\boldsymbol{X} = Y_i|_{\mathbf{Z}=\boldsymbol{z},R_{<i}=r_{<i}}$ yields that

$$\mathsf{Pr}_{\mathbf{t}=(\boldsymbol{z},r_0,r_1,y_1,\ldots,r_m,y_m)\xleftarrow{R}\widetilde{\mathbb{T}}}\left[\mathrm{H}_{Y_i|\mathbf{Z},R_{<i}}(y_i|\boldsymbol{z},r_{<i}) > d+\sum_{j=1}^{v}\mathrm{H}_{Y_{i,j}|\mathbf{Z},R_{<i}}(y_{i,j}|\boldsymbol{z},r_{<i})\right] \leq 2^{-d} = \mathrm{neg}(n) \tag{67}$$

for every $i\in[m]$. Summing over all $i\in[m]$, we get that

$$\mathsf{Pr}_{\mathbf{t}=(\boldsymbol{z},r_0,r_1,y_1,\ldots,r_m,y_m)\xleftarrow{R}\widetilde{\mathbb{T}}}\left[\sum_{i\in[m]}\mathrm{H}_{Y_i|\mathbf{Z},R_{<i}}(y_i|\boldsymbol{z},r_{<i}) > md+\sum_{i\in[m]}\sum_{j\in[v]}\mathrm{H}_{Y_{i,j}|\mathbf{Z},R_{<i}}(y_{i,j}|\boldsymbol{z},r_{<i})\right] = \mathrm{neg}(n) \tag{68}$$

It follows that

$$\mathsf{Pr}_{\mathbf{t}=(\boldsymbol{z},r_0,r_1,y_1,\ldots,r_m,y_m)\xleftarrow{R}\widetilde{\mathbb{T}}}\left[\sum_{i\in[m]}\sum_{j\in[v]}\mathrm{H}_{Y_{i,j}|\mathbf{Z},R_{<i}}(y_{i,j}|\boldsymbol{z},r_{<i}) \geq k'_{\mathrm{ACC}} - m\cdot d\right] \geq \varepsilon - \mathrm{neg}(n),$$

and therefore

$$\mathsf{Pr}_{\mathbf{t}=(\boldsymbol{z}.r_0,r_1,y_1,\ldots,r_m,y_m)\xleftarrow{R}\widetilde{\mathbb{T}},j\xleftarrow{R}[v]}\left[\sum_{i\in[m]}\mathrm{H}_{Y_{i,j}|\mathbf{Z},R_{<i}}(y_{i,j}|\boldsymbol{z},r_{<i}) > k_{\mathrm{ACC}} = (k'_{\mathrm{ACC}} - m\cdot d)/v\right] \geq \varepsilon - \mathrm{neg}(n) \tag{69}$$

Consider the following efficient G-consistent generator.

Algorithm A.10 (Generator $\widetilde{\mathsf{G}}$)**.**

Input: public parameter $z\in\{0,1\}^c$.

Operation:

1. *Sample* $j\xleftarrow{R}[v]$. *Let* $\boldsymbol{z}=(z_1,\ldots,z_w)$ *for* $z_v=z$, *and* z_i, *for* $i\neq v$, *sampled uniformly in* $\{0,1\}^c$.

   We will refer to the part of $\boldsymbol{z}$ *sampled by the generator as* $\boldsymbol{z}_{-v}$, *and (abusing notation) assume* $\boldsymbol{z}=(z,\boldsymbol{z}_{-v})$.

2. *Start a random execution of* $\widetilde{\mathbb{G}}(\boldsymbol{z})$ *and output the* j*'th entry of each output block.*

It is clear that $\widetilde{\mathsf{G}}$ is indeed an efficient G-consistent generator. We will show that the accessible entropy of $\widetilde{\mathsf{G}}$ violates the assumed bound on the accessible entropy of G.

Let $(Z,R'_0=(R_0,J,\mathbf{Z}_{-J}),R'_1,Y'_1,\ldots,R'_m,Y'_m)=T_{\widetilde{\mathsf{G}}}$. It is easy to verify that

$$\mathrm{H}_{Y'_i|Z,R'_{<i}}(y_i|z,(r_0,j,\boldsymbol{z}_{-j}),r_1,\ldots,r_{i-1}) = \mathrm{H}_{Y_{i,j}|\mathbf{Z},R_{<i}}(y_i|\boldsymbol{z}=(z,\boldsymbol{z}_{-j}),r'_{<i}=(r'_0.r_1,\ldots,r_{i-1})) \tag{70}$$

for every $(z, r'_0 = (r_0, j, \boldsymbol{z}_{-j}), r_1, y_1, \ldots, r_m, y_m) \in \mathrm{Supp}(T_{\widetilde{\mathsf{G}}})$ and $1 < i \leq m$. Thus, Equation (69) yields that

$$\mathsf{Pr}_{\mathbf{t} \xleftarrow{\mathrm{R}} T_{\widetilde{\mathsf{G}}}} \left[\mathrm{AccH}^{\widetilde{\mathsf{G}}}(\mathbf{t}) > k_{\mathrm{ACC}} \right] \geq \varepsilon - \mathrm{neg}(n).$$

The assumption about the inaccessible entropy of G yields that ε is negligible in n, and the proof of the lemma follows. □